MASTER'S THESIS

Bezint eer ge begint

Inzicht in het maken van de adoptiebeslissing voor Continuous Delivery

Houwaard, J. (Jeroen)

Award date:
2021

Bezint eer ge begint
Inzicht in het maken van de adoptiebeslissing voor Continuous Delivery

Opleiding:	Open Universiteit, faculteit Management, Science & Technology Masteropleiding Business Process Management & IT
Programme:	Open University of the Netherlands, faculty of Management, Science & Technology Master Business Process Management & IT
Cursus:	**IM9806 Afstudeeropdracht Business Process Management and IT**
Student:	Jeroen Houwaard
Identiteitsnummer:	
Datum:	18 januari 2021
Afstudeerbegeleider	Dhr. dr. Tibert Verhagen
Status:	Definitief

Abstract

In dit onderzoek staat centraal hoe organisaties de beslissing maken, al dan niet over te gaan tot de implementatie van Continuous Delivery. Hiervoor is allereerst onderzocht wat Continuous Delivery is, wat resulteert in tien componenten en zeven voordelen die in de bestaande literatuur gevonden zijn. Daarna is het implementatieproces van Continuous Delivery in kaart gebracht met behulp van het IT implementatieprocesmodel.
Uit de bestaande literatuur zijn vijf factoren geïnterpreteerd, waarvan verondersteld wordt dat zij van invloed zijn op het maken van de adoptiebeslissing.
Vervolgens is onderzocht hoe organisaties de adoptiebeslissing maken, waarbij de gevonden componenten en voordelen en de geïnterpreteerde determinanten getoetst zijn aan de empirie. Hieruit blijkt dat de voordelen van Continuous Delivery hun weg gevonden lijken te hebben naar de praktijk, maar de componenten in mindere mate. De grootste opbrengst is de empirische bevestiging die gevonden is voor de geïnterpreteerde determinanten klant, domein, architectuur, cultuur en testen. Organisaties lijken zich niet erg bewust te zijn van randvoorwaardelijke zaken bij de beslissing om tot Continuous Delivery over te gaan. Zo lang organisaties in het algemeen voldoende aan vernieuwing doen, lijkt Continuous Delivery eerder iets waar men als vanzelf inrolt. Een IT-push komt daarbij relatief vaak voor.

Sleutelbegrippen

Continuous Delivery, continous practices, software, implementatieproces, adoptiebeslissing, determinanten

Samenvatting

Sinds de publicatie van het Agile Manifesto heeft agile softwaredevelopment (ASD) een enorme vlucht genomen. Een relatief nieuwe ontwikkeling binnen deze stroming is het fenomeen Continuous Delivery. Deze practice streeft ernaar om nieuwe aanpassingen op software automatisch te testen en te deployen, teneinde een softwareproduct in continue staat klaar voor productie te hebben. Dit past in de paradigmaverschuiving, waarbij software steeds meer gezien wordt als zeer dynamisch product dat real-time in de behoeften voorziet.
In dit onderzoek wordt ingegaan op wat Continuous Delivery is en wat de verschillen zijn met Continuous Integration en Continuous Deployment. Hierbij zijn tien componenten gevonden: Snelle en frequente release, Flexibel productontwerp en architectuur, Continuous testing en quality assurance, Automatisering, Configuratiemanagement, Betrokken klant, Continue en snelle uitvoering van experimenten, Post-deployment activiteiten, Agile en Lean, Organisatorische factoren.
Daarnaast zijn zeven voordelen gevonden die Continuous Delivery kan opleveren: Kortere time-to-market, Continue feedback, Verbeterde betrouwbaarheid en kwaliteit van de release, Verhoogde klanttevredenheid, Verbeterde ontwikkelingsproductiviteit, Snelle innovatie, Smallere testfocus.
Het hoofddoel van het onderzoek is om erachter te komen hoe organisaties de beslissing maken om over te gaan tot de implementatie van Continuous Delivery en welke factoren van invloed zijn op het maken van die beslissing. Hiervoor wordt het IT implementatieprocesmodel gebruikt, met de focus op de eerste twee stappen: initiatie en adoptie. Bij initiatie wordt een (IT) oplossing geïdentificeerd als antwoord op een bepaalde kans. In dit geval bestaat die kans uit de voordelen die Continuous Delivery biedt. De stap van adoptie resulteert in het maken van de adoptiebeslissing.
Uit de bestaande literatuur zijn factoren geïnterpreteerd die van invloed zouden kunnen zijn op het maken van de adoptiebeslissing. Deze vijf determinanten zijn: klant, domein, architectuur, cultuur en testen.
In het empirisch onderzoek is ingegaan op hoe organisaties de adoptiebeslissing maken of hebben gemaakt. De gevonden componenten en voordelen zijn betrokken bij het onderzoek. Interessanter is het toetsen van de vijf determinanten aan de empirie. Deze determinanten zijn door de onderzoeker uit de bestaande literatuur geïnterpreteerd, waardoor de behoefte aan bevestiging en verbetering sterker is.
Uit het empirisch onderzoek blijkt dat de onderzochte organisaties zich erg bewust zijn van de voordelen van Continuous Delivery, maar in mindere mate van de componenten. Organisaties lijken dus goed te weten waarom ze de werkwijze willen toepassen, maar minder goed wat dat precies inhoudt. Er is bevestiging gevonden voor de geïnterpreteerde determinanten, al verschilt de mate waarin. Cultuur wordt het sterkst naar voren gebracht en testen het minst. Van de drie overzichten (voordelen, componenten, determinanten) wordt door de deelnemers ingeschat dat ze nuttig kunnen zijn in de praktijk, ter ondersteuning van organisaties bij het maken van de adoptiebeslissing.
Er komen ook nieuwe inzichten naar voren. Zo lijkt de beslissing om conform Continuous Delivery te gaan werken, niet bewust gemaakt te worden. Organisaties die in het algemeen meegaan met ontwikkelingen en vernieuwen, lijken als vanzelf de werkwijze te (kunnen) omarmen. Hierbij komt een IT-push relatief vaak voor in de onderzochte organisaties.

Summary

Ever since the Agile Manifesto was published, agile software development (ASD) has taken off. A relatively new development within this movement is the phenomenon of Continuous Delivery. This practice aims to automatically test and deploy new adaptations to software, in order to keep a software product in a releasable state (being able to deploy to production) at all times. This fits into the paradigm shift, where software is increasingly seen as a highly dynamic product that adapts real-time to business needs.
This research examines what Continuous Delivery is and it's differences compared to Continuous Integration and Continuous Deployment. Ten components were found:
Fast and frequent release, Flexible product design and architecture, Continuous testing and quality assurance, Automation, Configuration management, Customer involvement, Continuous and rapid experimentation, Post-deployment activities, Agile and Lean, Organizational factors.
In addition, seven benefits were found that Continuous Delivery can deliver: Shorter time-to-market, Continuous feedback, Improved release reliability and quality, Increased customer satisfaction, Improved development productivity, Rapid innovation, Narrower test focus.
The main objective of this study is to find out how organizations make the decision to adopt Continuous Delivery and which factors influence that decision. For this purpose the IT implementation process model is used, with a focus on the first two steps: initiation and adoption. At initiation, a(n IT) solution is identified as a response to a particular opportunity. In this case, that opportunity is found in the benefits of Continuous Delivery. The step of adoption results in making the decision to adopt and start implementation.
Factors have been interpreted, from the existing literature, that could influence the making of the adoption decision. These five determinants are: customer, domain, architecture, culture and testing. During empirical research, the researcher examined how organizations make, or have made the adoption decision. The components and benefits that were found, were included in the research, so their role can be determined. Even more interesting is testing the five determinants against empiricism. These determinants have been interpreted by the researcher from the existing literature, which suggests a stronger need for confirmation and improvement.
The empirical research shows examined organizations that are very much aware of the benefits of Continuous Delivery, but are in lesser extent aware of the components. Organizations seem to know just fine why they want the practice, but know less well what it implies exactly. Confirmation was found for the interpreted determinants, though in various extent. Confirmation was strongest on culture and weakest on testing. The three overviews (benefits, components, determinants) are assumed to be useful in practice by the participants, when it comes to making the adoption decision, they could support organizations.
This research also brings new insights. It seems the adoption decision to embrace the practice, isn't made in a consciously way. Organizations that evolve and renew in general, seem to almost automatically be able to implement Continuous Delivery. For the examined organizations, an IT-push was relatively often found.

Voorwoord

Hoe leuk het ook mocht zijn om nieuwe kennis op te doen, ben ik blij en opgelucht dat het schrijven van deze scriptie tot een einde komt. Ik ben doorgaans snel overtuigd van mijn eigen gelijk en kan moeiteloos standpunten van anderen naast mij neer leggen. Daarom heb ik de nodige moeite gehad om de zorgvuldigheid en diepgang die de wetenschap vereist, toe te voegen aan de door mijzelf opgedane kennis. Ik hoop van harte dat dat gelukt is en vind zelf dat het resultaat er mag zijn. Deze scriptie is tot stand gekomen naast mijn werk in een uitdagende functie en te midden van een jong gezin. Met name voor mijn gezin was dat niet altijd even leuk. Ik kijk er dan ook naar uit om nu eindelijk meer tijd en energie vrij te kunnen maken voor de mensen om mij heen. Zonder de hulp en begrip van mijn gezin en andere familie had ik dit niet kunnen doen en ik ben deze mensen dan ook erg dankbaar. Daarnaast wil ik ook mijn begeleider bedanken om zijn kritische houding aan het begin van dit traject. Ik kon het destijds waarschijnlijk niet zo waarderen als nu, maar het heeft (inderdaad) geholpen bij een soepel vervolg van het onderzoek.

Inhoudsopgave

1. Introductie

In dit eerste hoofdstuk wordt het onderwerp Continuous Delivery, samen met de andere kernbegrippen geïntroduceerd. Het doel is om de kernbegrippen te plaatsen in de wetenschappelijke literatuur en een heldere opdracht af te bakenen, waarbij duidelijk wordt welke kennis wordt toegevoegd.

1.1. Achtergrond

In de markt voor softwareontwikkeling is men op zoek naar manieren om software te leveren aan gebruikers, die ook daadwerkelijk de gebruikersbehoefte vervult. Met de opkomst van Agile softwareontwikkeling komt de focus meer te liggen op kortcyclische opleveringen van relatief kleine iteraties, teneinde grip te krijgen op resources en de waarde van de software. In 2010 wordt Continuous Delivery geïntroduceerd (Humble & Farley, 2010). Deze aanpak maakt het mogelijk om de tijd die het kost om nieuwe software van de ontwikkelaar naar de gebruiker te krijgen, sterk in te korten. Door testwerkzaamheden en het uitrollen van nieuwe software te automatiseren wordt het mogelijk met iedere (kleine) wijziging een nieuwe release (versie) van de software beschikbaar te stellen.

De ontwikkeling van Continuous Delivery kan worden geplaatst in een paradigmaverschuiving (Rodríguez et al., 2017). De traditionele blik op software als statisch product dat hoogstens iedere paar maanden wordt aangepast, maakt hierbij plaats voor het idee dat software zeer adaptief en waardegedreven is en zodoende een meer directe bijdrage aan de bedrijfsvoering levert. Rodríguez et al. (2017) leggen uit hoe Continuous Delivery een volgende logische stap is binnen Agile Software Development (ASD). De basis voor ASD werd gelegd met het Agile Manifesto (Beck et al., 2001), bestaande uit vier kernwaarden en twaalf principes. Wanneer we kijken naar het eerste principe: *'Our highest priority is to satisfy the customer through early and continous delivery of valuable software'*, wordt duidelijk dat sprake is van een letterlijke relatie tussen ASD en Continuous Delivery. Dat ASD streeft naar snelle levering van software komt ook terug in een ander principe: *'Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale'.* Continuous Delivery streeft ernaar de frequentie dusdanig te verhogen, dat sprake is van continue levering. Rodríguez et al. (2017) stellen dat Continuous Delivery als fenomeen daarmee verder gaat dan ASD, maar het kan ook gezien worden als een logische ontwikkeling binnen ASD. Continous Delivery versterkt ASD namelijk, het conflicteert niet.

Er wordt gesteld dat de industrie Continuous Delivery nog maar mondjesmaat omarmt (Laukkanen, Itkonen, & Lassenius, 2017), ondanks dat instructies voor implementatie al geruime tijd bestaan. De vraag ontstaat of men achterloopt, of dat implementatie meer moeite kost en minder oplevert dan voorstanders zeggen. Toch is ook duidelijk dat er interesse is in het onderwerp Continuous Delivery en hebben veel bedrijven plannen voor adoptie (Chen, 2017).

Laukkanen et al. (2017) stellen dat hun onderzoek naar de implementatieproblematiek zich uiteindelijk richt op de uitdaging van het maken van de adoptiebeslissing. De beslissing al dan niet over te gaan tot implementatie van Continuous Delivery en in welke mate. Het onderzoek richt zich echter niet op de fase waarin de beslissing wordt gemaakt.

Het is niet ondenkbaar dat het niet altijd verstandig om aan implementatie te beginnen, of dat verschillen in context tot verschillen in overweging en beslissing leiden. Inzicht in de overwegingen van organisaties en het maken van de adoptiebeslissing -al dan niet over te gaan tot de implementatie van Continuous Delivery- moet uitwijzen of de kennis over Continous Delivery in bestaande literatuur, ook zijn weg heeft weten te vinden naar het moment waarop de adoptiebeslissing wordt gemaakt. Ik wil daarom een onderzoek uitvoeren dat ingaat op hoe organisaties de adoptiebeslissing voor Continuous Delivery maken.

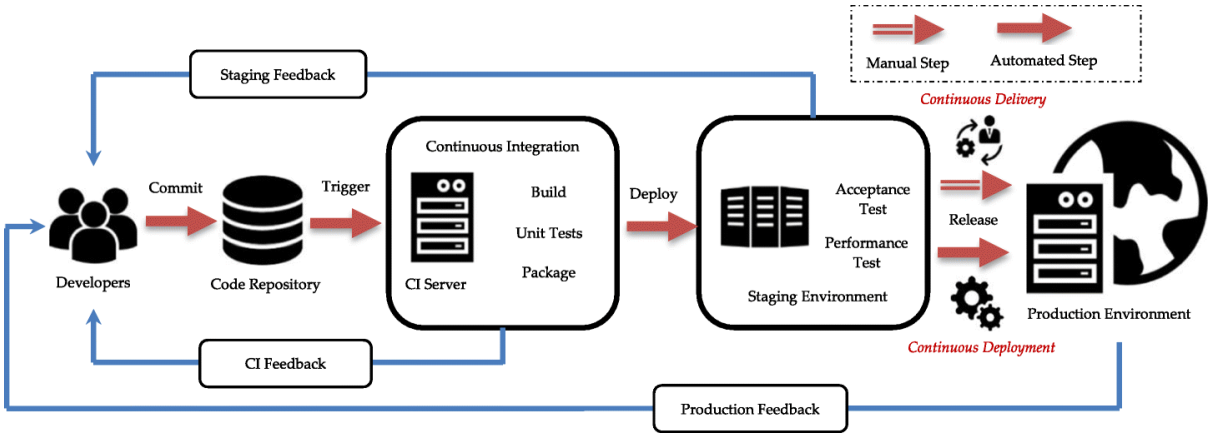
1.2. Gebiedsverkenning

Continuous Delivery wordt steeds bekender binnen het landschap van softwareontwikkeling (Chen, 2017). Het is een van verschillende practices die ingezet kunnen worden om het ontwikkelen van software te versnellen. Humble and Farley (2010) presenteren voor het eerst de practice, waarna het in steeds sterkere mate door de wetenschap wordt opgepakt. Achtereenvolgens zien we dat in de literatuur wordt ingegaan op de voordelen die Continuous Delivery biedt (Chen, 2015a), de uitdagingen die bij de implementatie komen kijken (Chen, 2015a, 2017; Neely & Stolt, 2013) en hoe om te gaan met die uitdagingen (Chen, 2017; Laukkanen et al., 2017). Rodríguez et al. (2017) stellen echter dat een gestructureerde definitie van de eigenschappen van Continuous Delivery nog niet bestaat en trachten die te bieden. Eveneens worden in dat onderzoek voordelen en uitdagingen geïdentificeerd. Een overzicht van de status van Continuous Delivery, inclusief de uitdagingen, hoe daarmee om te gaan en welke tools daarbij gebruikt worden, wordt gegeven door Shahin, Babar, and Zhu (2017). De hierboven genoemde literatuur laat daarmee een trend zien in onderwerpen met betrekking tot Continuous Delivery:

1. Karakteristieken (componenten van Continuous Delivery)
2. Voordelen (van Continuous Delivery)
3. Uitdagingen (bij implementatie van Continuous Delivery)
4. Oplossingen (hoe om te gaan met de eerder vastgestelde uitdagingen)
5. Tools (die helpen met het oplossen van de uitdagingen)

We zien verder dat de verschillen tussen Continuous Delivery, Continuous Integration en Continuous Deployment aan bod komen, waarbij ze gezamenlijk gepresenteerd worden als de continuous practices (Shahin et al., 2017).
Figuur 1 laat een model zien van een softwareontwikkelproces waarin de verschillen tussen de practices duidelijk worden (Shahin, Zahedi, Babar, & Zhu, 2019).
Verschillende ontwikkelaars werken aan een softwareproduct. Iedere nieuwe ontwikkeling dient geïntegreerd te worden in het bestaande product. Naarmate dit frequenter gedaan wordt, blijft de schade van een verkeerde ontwikkeling beperkter.
Continuous Integration (CI) streeft naar het continue integreren van nieuwe software. CI is wijdverbreid omarmt door de software-industrie (Shahin et al., 2017).
Continuous Delivery gaat een stap verder door in te zetten op geautomatiseerd deployen (uitrollen naar een volgende omgeving) en acceptatietesten. Het doel is om de software continue in een staat klaar voor release te hebben, zodat de software desgewenst naar een productieomgeving uitgerold kan worden (Shahin et al., 2017).
Continuous Deployment voegt daar een laatste stap aan toe door ieder stukje nieuwe software ook daadwerkelijk (geautomatiseerd) naar een productieomgeving uit te rollen (Shahin et al., 2017).
De continuous practices worden vaak afgekort tot CI, CD en CD. Hierbij laten de laatste twee afkortingen echter geen onderscheid zien. Shahin et al. (2017) gebruiken daarom CDE voor Continuous DElivery, maar ook daarmee wordt verwarring niet voorkomen, omdat het evengoed op Continuous DEployment kan slaan. In dit onderzoek houd ik derhalve CDV voor Continous DeliVery en CDP voor Continous DePloyment aan.
Binnen het vakgebied worden de termen vaak door elkaar gebruikt. CI kan bijvoorbeeld gebruikt worden, terwijl uit de context van de publicatie blijkt dat het om CDV gaat. Laukkanen et al. (2017) geven hier uitleg over, inclusief een voorbeeld waarbij de term CI wordt gebruikt, terwijl ook CDV gebruikt had kunnen worden (Eck, Uebernickel, & Brenner, 2014). Een ander voorbeeld blijkt uit Rodríguez et al. (2017), waar CDV in de titel wordt gebruikt, terwijl uit de context blijkt dat geen onderscheid wordt gemaakt tussen CDV en CDP. Bovendien merken Rodríguez et al. (2017) op dat in de meeste wetenschappelijke literatuur de twee termen verwisselbaar zijn.

De door Shahin et al. (2017) voorgestelde definities van de drie practices zouden een einde kunnen maken aan het verwarrende gebruik van de terminologie.

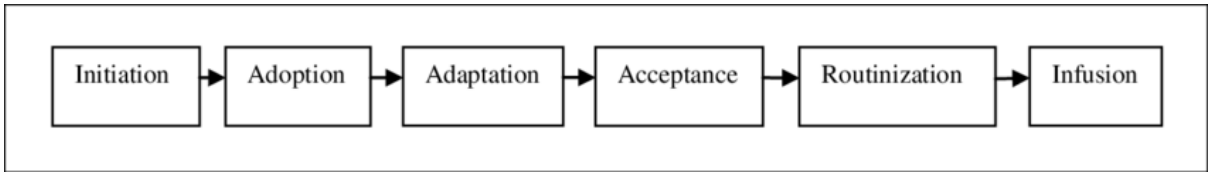


Figuur 1: The relationship between Continuous Integration, Delivery and Deployment (Shahin et al., 2019, p. 2)

De continuous practices vallen binnen het landschap van softwareontwikkeling. Meer specifiek gaat de ontwikkeling ervan hand in hand met de ontwikkeling van Agile softwareontwikkeling (ASD) (Rodríguez et al., 2017). Belangrijk om te beseffen is dat het niet gaat om ASD methoden als Srum of DSDM. De continuous practices staan los van dergelijke methoden of frameworks en kunnen in principe parallel worden toegepast.
Rodríguez et al. (2017) stellen dat sprake is van een verschuivend paradigma, waarbij software niet langer wordt gezien als statisch product dat zo nu en dan van een update wordt voorzien, maar als een product dat continue verandert en real-time aan de behoeften voldoet. De ontwikkeling van CDV en CDP kunnen niet los gezien worden van CI en andere ontwikkelingen die bijdragen aan de verschuiving.
Toch is het terecht om het fenomeen van CDV apart te benaderen. CI gaat puur om het ontwikkelen van software, zonder dat daar noodzakelijkerwijs het gebruikersperspectief bij betrokken is. CDV en CDP zijn in dat opzicht completer en impliceren het gebruikersperspectief, omdat niet alleen het proces van bouwen en testen van software, maar ook de acceptatie ervan wordt geautomatiseerd. Juist de dynamiek tussen ontwikkelaar en gebruiker is kritiek bij het ontwikkelen van software en komt terug in de Agile principes (Beck et al., 2001). Logischerwijs zijn de uitdagingen bij implementatie van geheel andere orde, omdat de kans groter is dat over verschillende organisaties samengewerkt moet worden, of dat feedback uit de markt komt.
Duidelijk is dat CDV niet op zichzelf staat en ontwikkeld is binnen het kader van ASD. In dit onderzoek zal CDV zoveel mogelijk apart benaderd worden. CI legt de basis voor CDV en kan niet genegeerd worden, maar andere ontwikkelingen binnen het landschap als Rapid Releases, Release Engineering en Test Driven Development worden buiten beschouwing gelaten.

1.3. Probleemstelling

Bestaand onderzoek naar CDV gaat hoofdzakelijk in op de eigenschappen, voordelen, uitdagingen, oplossingen en tools van CDV (Laukkanen et al., 2017; Rodríguez et al., 2017; Shahin et al., 2017). De onderzoeken richten zich met name op situaties waarbij CDV reeds geïmplementeerd is. Het doel van deze onderzoeken is dan ook om steeds beter te begrijpen wat CDV is en hoe CDV geïmplementeerd dient te worden.
Hoe organisaties komen tot de beslissing om over te gaan op het implementeren van CDV, is iets dat in de literatuur nog onderbelicht is gebleven. Dit wordt duidelijk in het onderzoek van Eck et al. (2014), waarbij wordt ingegaan op de drie laatste stadia van implementatie: acceptatie, routinisering en infusie. Hier wordt gebruik gemaakt van het model van het IT implementatie proces (Cooper & Zmud, 1990). Figuur 2 laat dat model zien met de zes stappen van dit proces. De eerste drie stappen kunnen getypeerd worden als pre-implementatie en de laatste drie stappen als post-implementatie (Eck et al., 2014).
Ook van eerder genoemd werk over de voordelen, uitdagingen, oplossingen en tools is evident dat het om post-implementatie studies gaat. Hierbij wordt de pre-implementatie situatie steeds buiten beschouwing gelaten. En juist daar zou een ongelukkige of matig onderbouwde uitvoering veel post-implementatie problematiek kunnen voorkomen (Laukkanen et al., 2017). Bestaand onderzoek biedt inzichten die kunnen ondersteunen bij het maken van de adoptiebeslissing, maar we weten niet of die inzichten ook ingezet worden bij het maken van de adoptiebeslissing (al dan niet over te gaan tot implementatie van CDV).
Alvorens een organisatie het besluit maakt om tot implementatie van CDV over te gaan, wordt een afweging gemaakt. De drie stadia van pre-implementatie zijn: initiatie, adoptie en adaptatie. De processtap adoptie leidt tot een resultaat: de beslissing om de investering te doen en tot implementatie over te gaan (Cooper & Zmud, 1990). De zes stadia kunnen als generiek worden beschouwd, waarbij we dus meer te weten willen komen over adoptie.



Figuur 2: IT implementation proces (Cooper & Zmud, 1990)

In het beschikbare onderzoek wordt tot nu toe steeds uitgegaan van een CDV implementatie (Laukkanen et al., 2017; Rodríguez et al., 2017; Shahin et al., 2017), terwijl onduidelijk is hoe organisaties tot de kritieke adoptiebeslissing komen. Wordt de kennis over CDV uit de bestaande literatuur gebruikt door organisaties op het moment dat ze de adoptiebeslissing maken?
Dit leidt tot de volgende probleemstelling:

Hoe maken organisaties de adoptiebeslissing, al dan niet over te gaan tot de implementatie van CDV, en welke factoren zijn hierbij van invloed?

In deze scriptie zal ik hier onderzoek naar doen. Uit de onderwerpen die bestaande literatuur biedt over CDV (eigenschappen, voordelen, problemen, oplossingen, tools) kunnen determinanten worden geïdentificeerd. Vervolgens kunnen deze determinanten worden getoetst in empirisch onderzoek.

1.4. Opdrachtformulering

Er wordt in dit onderzoek ingegaan op hoe organisaties de adoptiebeslissing (al dan niet over te gaan tot CDV implementatie) maken en welke overwegingen daarbij gemaakt worden. Er is kennis over de eigenschappen, voordelen, problemen, oplossingen en tools van CDV. Deze kennis dient geanalyseerd en gestructureerd te worden, zodat duidelijk wordt hoe deze kennis kan bijdragen aan de adoptiebeslissing. Dit wordt bereikt door factoren die invloed hebben op de adoptiebeslissing te identificeren uit de bestaande literatuur. Vervolgens zal middels empirisch onderzoek getoetst worden in hoeverre deze determinanten inderdaad gebruikt worden bij het maken van de adoptiebeslissing. Empirisch onderzoek kan eveneens andere determinanten toevoegen of eerder geïdentificeerde determinanten ter discussie stellen.

Onderzoeksvragen:

1. Welke factoren zijn van invloed bij het maken van de adoptiebeslissing, al dan niet tot implementatie van CDV over te gaan?
    a. Wat is CDV?
    b. Hoe ziet het implementatieproces van CDV eruit?
    c. Welke factoren kunnen we identificeren uit de bestaande literatuur, die mogelijk van invloed zijn bij het maken van de adoptiebeslissing?
2. Welke overwegingen maken organisaties bij het maken van de adoptiebeslissing, al dan niet tot implementatie van CDV over te gaan?
    a. Kan van de geïdentificeerde determinanten ook in de praktijk worden vastgesteld dat zij een rol spelen bij het maken van de adoptiebeslissing?
    b. Kunnen in de praktijk ook andere determinanten worden geïdentificeerd?

1.5. Motivatie / relevantie

Ondanks dat de bestaande kennis over CDV zou kunnen bijdragen aan de kwaliteit van de adoptiebeslissing, is deze kennis nog niet eerder gestructureerd voor dit specifieke doel. Voor organisaties die in de toekomst de adoptiebeslissing zullen maken, zal het zeer waardevol zijn om deze kennis aangereikt te krijgen.
Shahin et al. (2017) geven aan dat het implementeren van CDV geen triviale opgave is. Daaruit kan afgeleid worden dat de beslissing dit al dan niet te doen, zorgvuldig genomen dient te worden. Het doel van hun onderzoek is om de bestaande kennis over de continuous practices zodanig te analyseren en synthetiseren, dat ze als richtlijn kunnen dienen bij implementatie. Hoewel Shahin et al. (2017) niet expliciet ingaan op de adoptiebeslissing, kan hieruit wel worden afgeleid dat de kennis uit hun onderzoek zou moeten bijdragen aan het maken van de adoptiebeslissing, aangezien dat een onderdeel is van het implementatieproces (Cooper & Zmud, 1990).
Laukkanen et al. (2017) geven aan dat CDV succesvol geïmplementeerd is bij vooroplopende organisaties, maar dat nog niet bekend is in hoeverre het algemeen toepasbaar is. Verschillende contexten vragen om verschillende manieren om CDV te implementeren. Het zou voor organisaties erg waardevol zijn wanneer zij begrijpen hoe CDV te implementeren voor hun eigen context.
Hoewel Laukkanen et al. (2017) stellen dat hun onderzoek zich richt op de adoptiebeslissing (*"We aim to address the decision-making challenge of wheter to adopt CD or not and to what extent"* (Laukkanen et al., 2017, p. 56)), komt dat niet terug in de onderzoeksvragen en de conclusies en blijft onduidelijk hoe de kennis direct moet bijdragen aan het maken van de adoptiebeslissing. We kunnen hier alleen de aanname doen dat belanghebbenden die betrokken zijn bij de adoptiebeslissing, zich kunnen inlezen in CDV met behulp van wetenschappelijke literatuur. Daarom is het zinvol om te toetsen in hoeverre de kennis uit de literatuur aanwezig is bij het maken van de

adoptiebeslissing. We weten nog niet in hoeverre de wetenschap slaagt in het doel het maken van de adoptiebeslissing te verbeteren.
Ik verwacht met dit onderzoek te laten zien welke invloed bepaalde contextuele factoren hebben en hoe dit meegewogen dient te worden bij het maken van de adoptiebeslissing. In het gunstigste geval kan het leiden tot vastgestelde determinanten, die organisaties kunnen helpen bij het maken van de adoptiebeslissing.

1.6. Aanpak in hoofdlijnen

De eerste onderzoeksvraag en deelvragen lenen zich uitstekend voor literatuuronderzoek en zullen beantwoord worden in hoofdstuk 2. De antwoorden op deze vragen worden verwerkt tot het ontwerp van het empirisch onderzoek in hoofdstuk 3.
De resultaten van het empirisch onderzoek worden getoond in hoofdstuk 4 en verder besproken in hoofdstuk 5 en geven antwoord op de tweede onderzoeksvraag.

2. Theoretisch kader

In dit hoofdstuk wordt het theoretisch kader uitgewerkt. Allereerst wordt ingegaan op de onderzoeksaanpak (2.1), vervolgens op de uitvoering (2.2). Het zwaartepunt van dit hoofdstuk ligt echter in het derde deel, de uitwerking van het resultaat van het literatuuronderzoek (2.3).

2.1. Onderzoeksaanpak

Het doel van het literatuuronderzoek is om kennis in de bestaande literatuur over CDV, op dusdanige wijze te analyseren en structureren, dat zij gebruikt kan worden voor empirisch onderzoek naar het maken van de adoptiebeslissing. Om dit te bereiken zullen factoren geïdentificeerd worden die van invloed zijn op het maken van de adoptiebeslissing. Daarnaast zal het literatuuronderzoek ook duiding moeten geven aan de adoptiebeslissing en het implementatieproces.
Dit komt ook terug in het de eerste onderzoeksvraag uit hoofdstuk 1.

Er is bij het zoeken naar geschikte literatuur gebruik gemaakt van de universiteitsbibliotheek van de Open Universiteit.
Zoals in de introductie ook benoemd, worden CDV en CDP soms allebei gebruikt met dezelfde betekenis (Laukkanen et al., 2017; Rodríguez et al., 2017). Bij het zoeken naar literatuur is hiermee rekening gehouden en beide zoektermen zijn gebruikt.
Wat bij een eerste verkenning van het onderwerp in de universiteitsbibliotheek opvalt, is dat CDV ook een fenomeen is in de geneeskunde, waar het slaat op het continu toedienen van medicatie. Het is evident dat deze twee fenomenen niets met elkaar te maken hebben en daarom is gekeken hoe in beperkter kader gezocht kan worden. Eerst heb ik gefilterd op vakgebied, maar daarbij bleven teveel irrelevante artikelen bestaan in het resultaat, tenzij dit ten koste ging van relevante artikelen. Door ‘software’ toe te voegen aan onderwerp termen, vallen de irrelevante resultaten buiten beschouwing en lijken de relevante resultaten niet verloren te gaan. Het boek van Humble and Farley (2010) is als uitgangspunt genomen, waardoor alleen gezocht is naar publicaties vanaf 2010. Zie verder tabel 1 voor details van de zoekcriteria.

Zoekterm: Titel	Continuous delivery
Zoekterm: Titel (OR)	Continuous deployment
Zoekterm: Termen onderwerp (AND)	Software
Query	((TitleCombined:(continuous delivery)) OR (TitleCombined:(continuous deployment))) AND (SubjectTerms:(software))
Publicatiedatum	Vanaf 1-1-2010
Beperkt tot	Peer-reviewed publicaties
Uitsluiten	Krantenartikelen, boekrecensies, niet Engelstalig

Tabel 1: Zoekcriteria literatuur

2.2. Uitvoering

Na verdieping in het onderwerp blijkt dat CDV voor het eerst beschreven is door Humble and Farley (2010), in een boek dat de complete practice uiteenzet. De uitgave is veelvuldig geciteerd, doorgaans in de introductie van het artikel en het onderwerp CDV (e.g.: Chen, 2015a, 2017; Laukkanen et al., 2017; Neely & Stolt, 2013; Rodríguez et al., 2017; Shahin et al., 2017).
De zoekcriteria van 2.1 leveren in eerste instantie 25 publicaties op. Na analyse van titel en abstract worden daarvan 14 publicaties als relevant beoordeeld. Zie hiervoor bijlage 1. Daarbij wordt de kern

gevormd door drie overzichtspublicaties, waarvan een systematic mapping study en twee systematic literature reviews:

- Rodríguez et al. (2017), SMS bevat 50 studies
- Laukkanen et al. (2017), SLR bevat 30 studies
- Shahin et al. (2017), SLR bevat 69 studies

Dit betreffen namelijk grondige (secundaire) studies, waarin wordt getracht een compleet beeld te geven van (onder andere) CDV.
Van de 11 publicaties die overblijven komen er 3 voor in minimaal een van bovengenoemde publicaties. De andere 8 zijn hiervoor waarschijnlijk te recent gepubliceerd. De publicaties die niet zijn meegenomen in de overzichtspublicaties zullen zorgvuldiger bestudeerd moeten worden dan het werk dat wel onderdeel is van de overzichtspublicaties.
Middels de sneeuwbalmethode (Verschuren & Doorewaard, 2007) is de overige literatuur toegevoegd aan de gebruikte literatuur. Wanneer bij beweringen uit de gevonden literatuur verwezen is naar bepaalde bronnen, zijn in verschillende gevallen die bronnen toegevoegd aan de gebruikte literatuur. Ik heb niet op systematische wijze de literatuurlijsten onderzocht. Een overzicht van de bronnen die zo gevonden zijn, is toegevoegd aan bijlage 1.

Een mogelijke tekortkoming in de gebruikte zoekcriteria (tabel 1) is het feit dat alleen op titel wordt gezocht, terwijl de termen ook in de samenvatting (abstract) van de publicatie kunnen voorkomen, zonder dat ze in de titel voorkomen. Daarom is in later stadium het zoeken naar literatuur uitgebreid op dit punt. Zie hiervoor tabel 2. Omdat weer andere publicaties werden gevonden bij het gebruik van 'Termen onderwerp', is deze ook toegevoegd aan de zoekquery. Tot slot bleek dat bij deze hoeveelheid resultaten eveneens een verschil ontstaat tussen gebruik van 'software' als onderwerpsterm en 'software' in de samenvatting. Daarom zijn beide gebruikt. Bij de zoekquery van tabel 1 leverde dit geen relevante verschillen op en is maar op één zoekterm beperkt.

Zoekterm: Titel	continuous delivery
Zoekterm: Titel (OR)	continuous deployment
Zoekterm: Samenvatting (OR)	"continuous delivery"
Zoekterm: Samenvatting (OR)	"continuous deployment"
Zoekterm: Termen onderwerp (OR)	"continuous delivery"
Zoekterm: Termen onderwerp (OR)	"continuous deployment"
Zoekterm: Termen onderwerp (AND)	software
Zoekterm: Samenvatting (OR)	software
Query	((TitleCombined:(continuous delivery)) OR (TitleCombined:(continuous deployment)) OR (Abstract:("continuous delivery")) OR (Abstract:("continuous deployment")) OR (SubjectTerms:("continuous delivery")) OR (SubjectTerms:("continuous deployment"))) AND ((SubjectTerms:("software")) OR (Abstract:("software")))
Publicatiedatum	Vanaf 1-1-2010
Beperkt tot	Peer-reviewed publicaties
Uitsluiten	Krantenartikelen, boekrecensies, niet Engelstalig

Tabel 2: uitgebreidere zoekcriteria literatuur

De query van tabel 2 gaf 85 resultaten en bevat al de resultaten van de eerste query (25). Van de overige 60 publicaties waren er negen evident irrelevant, bijvoorbeeld omdat het onderwerp toch gezondheidszorg was, ondanks de beperking op software. De overgebleven 51 publicaties zijn beoordeeld op abstract en bij twijfel op inhoud. Geen daarvan zijn als relevant beoordeeld voor dit onderzoek. Het doel hierbij was om te voorkomen een belangrijke publicatie te missen. Wanneer deze publicaties wel zouden zijn betrokken bij het onderzoek, konden ze mogelijk wel van enige

toegevoegde waarde zijn. Gezien de beperking in tijd en middelen voor dit onderzoek zijn de publicaties echter niet nader bestudeerd. Een overzicht van de betreffende uitgesloten publicaties is opgenomen in bijlage 2.

2.3. Resultaten en conclusies

De gevonden literatuur is gebruikt om antwoord te geven op de eerste onderzoeksvraag, opgebouwd uit drie deelvragen. Dit onderdeel is daarom in lijn met de deelvragen opgebouwd. In 2.3.1. wordt beschreven wat CDV is. Vervolgens gaat 2.3.2. in op het implementatieproces van CDV. Tot slot gaan 2.3.3. in op de geïdentificeerde determinanten. Reflectie op de gehele onderzoeksvraag komt aan bod in 2.4.

2.3.1. Wat is Continuous Delivery

Een eerste stap in het beschrijven wat CDV is, is het geven van een definitie. Ondanks dat door Humble and Farley (2010) compleet is uiteengezet wat CDV is, geven zij geen concrete definitie. Rodríguez et al. (2017) leveren een overzicht van zeven definities, met als doel antwoord te geven op de vraag wat onderzoekers bedoelen, wanneer zij de term CDV/CDP gebruiken. Hieruit blijkt dat één definitie, waar volledige overeenstemming over is bereikt, nog niet bestaat. Een van de auteurs van het initiële werk waarmee het concept werd geïntroduceerd (Humble & Farley, 2010), komt jaren later met deze introductie van het onderwerp:
'Continuous Delivery is a set of principles, patterns and practices designed to make deployments - whether of a large-scale distributed system, a complex production environment, an embedded system, or a mobile app- predictable, routine affairs that can be performed on demand at any time.' (Humble, 2017, p. 1)
De overeenstemming die bereikt wordt in de zeven definities (Rodríguez et al., 2017), komt terug in deze definitie van Humble (2017). Omdat deze definitie recenter is, gegeven is door de initiële auteur en geen probleem oplevert met betrekking tot het verschil tussen CDV en CDP, zie ik hem als meest geschikt voor dit onderzoek.
In de definitie wordt duidelijk dat iets gedaan moeten worden (*principles, patterns and practices*), om tot een bepaald resultaat (*to make deployments … routine affairs…*) te komen. Met CDV, bestaande uit bepaalde componenten, worden dus bepaalde voordelen behaald. Dit komt ook terug in het werk van Rodríguez et al. (2017). De zeven definities vormen de inleiding voor het bespreken van de componenten en voordelen van CDV/CDP[1]. Op deze wijze trachten Rodríguez et al. (2017) te duiden wat CDV nu precies is.

De componenten van CDV
De componenten worden gepresenteerd in tien hoofdthema's, die gezamenlijk het fenomeen karakteriseren (zie tabel 3). Het gaat hier om thema's die steeds terugkomen in de onderzochte literatuur en zodoende geïdentificeerd zijn als onderdeel dat bijdraagt aan de implementatie van CDV in de praktijk (Rodríguez et al., 2017).
Hieruit blijkt dat de componenten niet zuiver de eigenschappen van CDV beschrijven, maar ook al iets zeggen over implementatie. Automatisering draagt bijvoorbeeld bij aan het mogelijk maken van snelle en frequente releases. CDV is iets dat bereikt kan worden, niet iets dat eenvoudig aan of uit

[1] Rodríguez et al. (2017) maken geen onderscheid tussen CDV en CDP en gebruiken CD om het fenomeen te benoemen. Voorts gebruik ik CDV.

gezet wordt. Dit is in lijn met de hierboven genoemde definitie van Humble (2017), die de componenten samenvat tot een set van principes, patronen en practices.
De tien componenten (tabel 3) worden door Shahin et al. (2017) overgenomen en de hoofdbijdrage uit de studie van Rodríguez et al. (2017) genoemd. Ook Laukkanen et al. (2017) citeren de tien factoren, maar geven aan het fenomeen CDV als beperkter te zien. Zij zouden de definitie van het initiële werk (Humble & Farley, 2010) aanhouden, waarmee factoren 6 en 7 afvallen. Het probleem hierbij, is dat Humble and Farley (2010) in hun werk niet echt een definitie geven, maar wel aangeven dat zij op dat moment niet alle mogelijke voordelen (gevolgen) van hun practice kunnen overzien.

Component	Beschrijving
1. Snelle en frequente release	Het vermogen om software uit te rollen naar productie wanneer de organisatie wil, gebaseerd op de behoefte, maar gericht op een zo kort mogelijk cyclus (wekelijks of dagelijks), of zelfs continu.
2. Flexibel productontwerp en architectuur	CDV/CDP vereist een robuuste software architectuur met als doel een balans te vinden in snelheid en stabiliteit.
3. Continuous testing en quality assurance	De kwaliteit van de software is te allen tijde compromisloos verzekerd, ondanks snel en continu uitrollen naar ander omgevingen.
4. Automatisering	Het automatiseren van de delivery pipeline, van initiële software build, tot testen, verder uitrollen en monitoren.
5. Configuratiemanagement	Versiebeheer van alle code en configuraties.
6. Betrokken klant	Mechanismen om klanten betrokken te krijgen bij het ontwikkelproces en zodoende zo vroeg mogelijk feedback te krijgen, teneinde invloed uit te oefenen op ontwerp en innovatie.
7. Continue en snelle uitvoering van experimenten	Het systematisch ontwerpen en uitvoeren van kleine experimenten die vervolgens de productontwikkeling gidsen en innovatie versnellen.
8. Post-deployment activiteiten	Activiteiten die worden uitgevoerd nadat het product (of uitbreiding of verbetering ervan) uitgerold is, om snelle beslissingsvorming te ondersteunen.
9. Agile en Lean	Verdere uitbreiding op Agile en Lean softwareontwikkeling.
10. Organisatorische factoren	Organisatorische factoren die CDV/CDP mogelijk maken.

Tabel 3: De componenten van CDV (Rodríguez et al., 2017, p. 273)

Het uitsluiten van factoren 6 en 7 is een keuze van Laukkanen et al. (2017). Mogelijk zien zij de factoren uitsluitend als onderdeel van CDP, maar dat wordt niet benoemd. Men heeft het over het (uit)rekken van CDV. De drie secundaire studies lopen uiteen in hoe de begrippen CDV en CDP gezien worden, te zien in tabel 4, waardoor ze niet probleemloos naast elkaar gelegd kunnen worden en het belangrijk is om nauwkeurig te duiden hoe CDV gezien wordt in de context van dit onderzoek.

Secundaire studie	CDV, CDP
Rodríguez et al. (2017)	Geen onderscheid tussen CDV en CDP
Laukkanen et al. (2017)	Focus op CDV, CDP benoemd als uitbreiding op CDV
Shahin et al. (2017)	CDV en CDP worden als verschillend fenomeen beschreven

Tabel 4: Vergelijking van de secundaire studies

Figuur 1 (in hoofdstuk 1) laat een eenvoudig model zien, waarvan de essentie is dat op verschillende omgevingen, verschillende activiteiten worden uitgevoerd. Voordat software in gebruik genomen wordt, wordt een acceptatietest uitgevoerd op een acceptatieomgeving (staging environment figuur 1). Het installeren van de nieuwe software op de acceptatieomgeving (deploy figuur 1) gaat gepaard met risico en kost tijd. Automatisering tracht een einde te maken aan het risico en de tijdsduur drastisch in te korten. Voor installeren van de nieuwe software op de productieomgeving (release figuur 1) geldt bij CDP hetzelfde. Waar figuur 1 onderscheid maakt tussen deploy en release, komt het technisch op hetzelfde neer. Wie in staat is om nieuwe software geautomatiseerd uit te rollen naar een acceptatieomgeving, zal ook in staat zijn nieuwe software geautomatiseerd uit te rollen naar de productieomgeving. Derhalve is het slecht denkbaar dat een organisatie die CDV volledig geïmplementeerd heeft, de uitrol van nieuwe software naar de productieomgeving nog steeds handmatig zal uitvoeren. Het doel van CDV is om een softwareproduct continu in een staat te hebben, waarin het uitgerold kan worden naar de productieomgeving (Humble & Farley, 2010; Shahin et al., 2017). Het daadwerkelijk uitrollen naar de productieomgeving blijft een bewuste stap. Het verschil tussen CDV en CDP is dus niet zozeer het verschil tussen handmatig en automatisch uitvoeren van de release, maar meer het verschil in staat te zijn iedere softwarewijziging automatisch tot een nieuwe release te laten leiden en het ook daadwerkelijk altijd meteen doen. Dit wordt bevestigd in het werk van Dingsøyr and Lassenius (2016). De stap van CDV naar CDP is dus relatief klein, waarbij CDV niet betekent dat geen sprake kan zijn van CDP. In dit onderzoek staat daarom CDV centraal. Afhankelijk van de context van het softwareproduct, is het een bedrijfsmatige keuze om uiteindelijk voor CDV of CDP te gaan. Dit wordt bevestigd door de auteur van het initiële boek (Humble, 2010), waarop door Shahin et al. (2017) worden geconcludeerd dat CDV voor alle type systemen of organisaties toepasbaar is, terwijl CDP slechts toepasbaar is voor bepaalde systemen of organisaties (Shahin et al., 2017, p. 3911). Van de componenten in tabel 2 worden voor onderhavig onderzoek dus geen componenten uitgesloten, zoals door Laukkanen et al. (2017).

De voordelen die CDV oplevert

Naast de componenten van CDV, wordt in veel werk ook ingegaan op de voordelen die CDV moet brengen (e.g.:Chen, 2015a; Laukkanen et al., 2017; Leppanen et al., 2015; Rodríguez et al., 2017). Genoemde voordelen zijn: *shorter time-to-market, increased customer satisfaction, continuous feedback, rapid innovation, narrower test focus, improved release reliability and quality, improved developer productivity* (Rodríguez et al., 2017, p. 282). Maar ook: *automated acceptance and unit tests, improved communication, increased productivity, increased project predictability as an effect of finding problems earlier, increased customer satisfaction, shorter time-to-market, narrower test scope, improved release reliability and quality* (Laukkanen et al., 2017, p. 57)
Laukkanen et al. (2017) baseren de genoemde voordelen op drie secundaire studies, waaronder Rodríguez et al. (2017).
Wanneer we de genoemde voordelen vergelijken valt op dat ze in hoge mate overeenkomen (tabel 5).

(Laukkanen et al., 2017)		(Rodríguez et al., 2017)
automated acceptance and unit tests		
improved communication		
increased productivity	=	improved developer productivity
increased project predictability as an effect of finding problems earlier	≈	continuous feedback
increased customer satisfaction	=	increased customer satisfaction
shorter time-to-market	=	shorter time-to-market
narrower test scope	=	narrower test focus
improved release reliability and quality	=	improved release reliability and quality
		rapid innovation

Tabel 5: de voordelen van CDV vergeleken

In de vergelijking van de componenten was al opgevallen dat Laukkanen et al. (2017) het snel en continu uitvoeren van experimenten niet zuiver als onderdeel van CDV zien. Daarmee is het logisch dat ook snelle innovatie afvalt als voordeel.
De twee genoemde voordelen van Laukkanen et al. (2017), die niet genoemd worden door Rodríguez et al. (2017) (*automated acceptance and unit tests, improved communication*) zijn geen werkelijke voordelen. Wanneer een component van CDV correct tot uiting kan komen, kan het tot een voordeel leiden. Snelle en meer frequente releases leiden bijvoorbeeld tot een snellere time-to-market. De twee genoemde voordelen zijn in die zin niet echt voordelen, maar eerder eigenschappen. '*Automated acceptance and unit tests*' behoort tot de eigenschap '*automatisering*' in tabel 4. '*Improved communication*' behoort dan tot '*betrokken klant'* in tabel 4. Wanneer communicatie toch als (resultaat) voordeel en niet zozeer als middel gezien wordt, kan het gezien worden als '*continuous feedback*'. Dit wordt bevestigd in de beschrijving van het component '*betrokken klant*'.
De twee voordelen blijken overigens niet, zoals gesteld door Laukkanen et al. (2017), uit een SLR, maar uit een case study te komen (Ståhl & Bosch, 2013). Deze case study legt de focus op CI. Waar we in ander werk zien dat met CI ook CDV bedoeld kan worden (Eck et al., 2014), zou het kunnen dat Ståhl and Bosch (2013) daadwerkelijk CI bedoelen. Dit kan een verklaring zijn voor het feit dat deze voordelen niet in lijn zijn met ander gerapporteerde CDV voordelen. Laukkanen et al. (2017) stellen namelijk zowel de voordelen van CI als CDV te noemen in hun werk. Derhalve laat ik de twee voordelen voor dit onderzoek buiten beschouwing en steun ik op het werk van Rodríguez et al. (2017), wat leidt tot de voordelen in tabel 6.

Voordeel	Beschrijving
1. Kortere time-to-market	Een direct gevolg van snellere en meer frequente releases, is dat software sneller beschikbaar is op de markt.
2. Continue feedback	Door nieuwe software vaker en sneller uit te rollen naar de klant, wordt deze in staat gesteld vaker feedback te geven.
3. Verbeterde betrouwbaarheid en kwaliteit van de release	Het automatiseren van het testwerk en het uitrollen van nieuwe versies, zorgt ervoor dat de betrouwbaarheid en kwaliteit omhoog gaat.
4. Verhoogde klanttevredenheid	Klanten kunnen vaker en sneller feedback geven en deze feedback leidt tot oplossingen en verbeteringen.
5. Verbeterde ontwikkelingsproductiviteit	Doordat een ontwikkelaar een nieuwe verbetering na ontwikkeling direct beschikbaar kan maken en dit kan terugdraaien als dat nodig is.
6. Snelle innovatie	Snellere feedback en beter contact met klanten leidt tot snellere innovatie.
7. Smallere testfocus	Een nieuwe versie van de software bevat minimale wijzigingen in de code.

Tabel 6: De voordelen van CDV (Rodríguez et al., 2017)

Andere voordelen die benoemd worden in de literatuur zijn: *accelerated time to market, building the right product, improved productivity, reliable releases, improved product quality, improved customer satisfaction (Chen, 2015a).* Deze komen allen voor in de voordelen genoemd in tabel 5, eventueel in iets andere woorden. Alleen het bouwen van het juiste product lijkt mogelijk anders, maar uit de beschrijvingen valt af te leiden dat deze valt onder continue feedback.
Siqueira, Camarinha, Wen, Meirelles, and Kon (2018) nemen de voordelen over van Chen (2015a). Leppanen et al. (2015) noemen de volgende voordelen: *faster feedback, more frequent releases, improved quality and productivity, improved customer satisfaction, effort savings, a closer connection between development and operations.* Ook hier komen de voordelen weer in hoge mate overeen met de door Rodríguez et al. (2017) gerapporteerde voordelen. Alleen met de laatste twee wordt mogelijk iets anders bedoeld. Het besparen van moeite is echter geen voordeel, maar valt onder het component '*automatisering*'. Omdat het testen en uitrollen van nieuwe software is geautomatiseerd en geen moeite meer kost, zijn organisaties in staat sneller en frequenter nieuwe software naar productie te brengen. Dit is de kern van CDV, waarbij het uiteindelijke voordeel het bereiken van een kortere time-to-market is, niet slechts het besparen van moeite. Een betere relatie tussen development en operations is eveneens geen voordeel, maar iets dat nodig is om CDV succesvol toe te passen. Dit komt terug in de component '*organisatorische factoren*' die Rodríguez et al. (2017) noemen. Derhalve laat ik de voordelen *effort savings, a closer connection between development and operations* buiten beschouwing en blijven de door Rodríguez et al. (2017) gerapporteerde voordelen gehandhaafd (tabel 6).

2.3.2. Het implementatieproces van CDV

Het hoofddoel in dit onderzoek is meer te weten te komen over hoe organisaties de adoptiebeslissing (al dan niet over te gaan tot implementatie van CDV) maken. Hierbij gebruik ik het IT implementation process (Cooper & Zmud, 1990) als model om CDV implementatie in te plaatsen. Cooper and Zmud (1990) typeren het IT implementatieproces als de organisatorische inspanning, gericht op het verspreiden van IT binnen een gebruikersgemeenschap. Hiermee lijkt het een

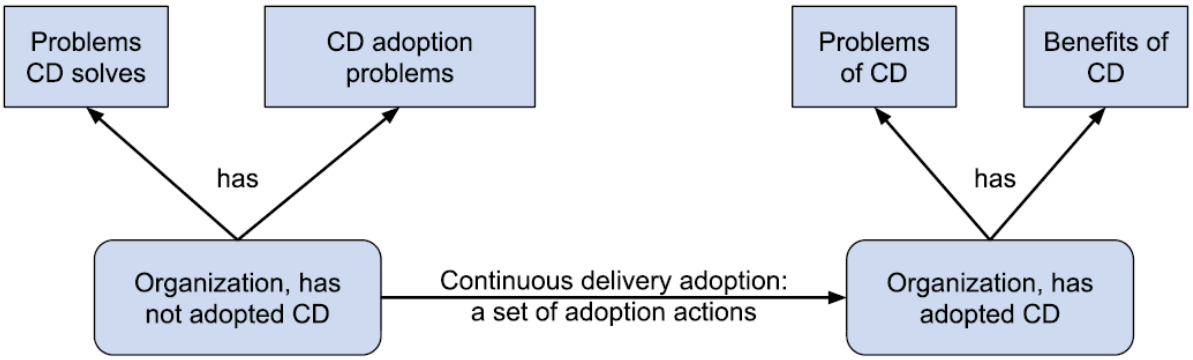
algemeen toepasbaar model, ook geschikt voor het in kaart brengen van de implementatie van CDV, ondanks dat de beschrijving van het model het over een systeem heeft. Daarnaast is het model eerder gebruikt om de implementatie van een continuous practice op te projecteren (Eck et al., 2014).
Het voordeel van dit model met betrekking tot deze studie is dat de adoptiebeslissing expliciet wordt genoemd als resultaat van de adoptiefase. Adoptie is dan ook een fase binnen het implementatieproces (zie tabel 7). Er is ook literatuur beschikbaar waar adoptie als term ook gebruikt wordt om het gehele implementatieproces aan te duiden.

Andere modellen die gebruikt worden bij het in kaart brengen van de implementatie (adoptie) van IT zijn het Technology-Organization-Environment (TOE) framework (Tornatzky & Fleischer, 1990) en de Diffusion of Innovations (DOI) theorie (Rogers, 2003). Deze modellen kennen niet de voordelen (expliciete adoptiebeslissing en eerder gebruikt in CDV context) die het IT implementation process (Cooper & Zmud, 1990) wel kent.

Fase	Proces	Product
Initiatie	Actief en/of passief scannen van organisatorische problemen/kansen en IT oplossingen wordt ondernomen. Druk om te veranderen ontwikkelt zich vanuit organisatorische behoefte (pull), technologische innovatie (push) of beide.	Een overeenkomst is gevonden tussen IT oplossing en de toepassing daarvan in de organisatie.
Adoptie	Rationele en politieke onderhandelingen volgen om organisatorische steun te krijgen voor de implementatie van de IT toepassing.	Een beslissing is gemaakt om de vereiste middelen te investeren teneinde de implementatie mogelijk te maken.
Adaptatie	De IT toepassing wordt ontwikkeld, geïnstalleerd en onderhouden. Organisatorische procedures worden herzien en ontwikkeld. Mensen binnen de organisatie worden getraind in zowel de nieuwe procedures als de IT toepassing.	De IT toepassing is klaar voor gebruik in de organisatie.
Acceptatie	Mensen in de organisatie worden ertoe bewogen zich te committeren aan het gebruik van de IT toepassing.	De IT toepassing is in gebruik genomen bij het werk binnen de organisatie.
Routinisering	Gebruik van de IT toepassing als normale gang van zaken wordt aangemoedigd.	De IT toepassing wordt niet meer als iets bijzonders gezien.
Infusie	Toegenomen organisatorische effectiviteit wordt bereikt, door de IT toepassing op een meer uitgebreide en geïntegreerde manier te gebruiken, om aspecten van een hoger niveau van organisatorisch werk te ondersteunen.	De IT toepassing wordt in zijn volledige potentieel binnen de organisatie gebruikt.

Tabel 7: IT implementation process (Cooper & Zmud, 1990, pp. 124-125)

Het IT implementatie model (Cooper & Zmud, 1990) is van generieke aard en kan specifieker worden gemaakt voor CDV. Hiervoor kijken we naar het onderscheid dat gemaakt wordt tussen de verschillende soorten problemen van CDV (Laukkanen et al., 2017).

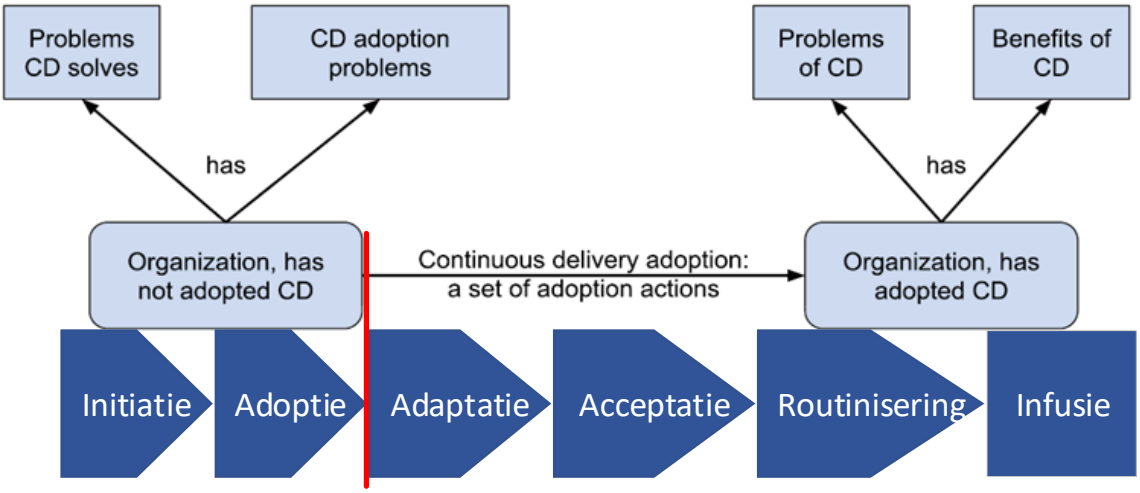


Figuur 3: Onderscheid tussen CDV (hier CD) problemen en voordelen (Laukkanen et al., 2017, p. 57)

Hierbij bestaan twee situaties, met ieder twee eigenschappen:

1. Een organisatie, CDV nog niet geïmplementeerd, beschikt over:
    a. Problemen die CDV oplost
    b. CDV implementatie problemen
2. Een organisatie, CDV reeds geïmplementeerd, beschikt over:
    a. Problemen die CDV oplevert
    b. Voordelen die CDV biedt

Zie hiervoor ook figuur 3, waar tevens te zien is dat een organisatie een set van implementatie acties moet uitvoeren, waarmee de transitie van de eerste naar de tweede situatie bereikt kan worden. In werkelijkheid is het echter niet zo eenvoudig en zijn ampel voorbeelden beschikbaar van organisaties die al snel drie tot vier jaar aan dergelijke implementatie moeten werken (Chen, 2017; Humble, 2017; Siqueira et al., 2018). De problemen die CDV oplost, zouden ook geformuleerd kunnen worden als het gebrek aan de voordelen van CDV. En andersom zijn de voordelen van CDV niets anders dan het niet hebben van de problemen die CDV oplost. Deze relatie tussen 1a en 2b laat zien dat het noodzakelijk is om voorafgaand aan de implementatie van CDV te weten wat de voordelen van CDV zijn. De eerste stap van het IT implementatieproces (Cooper & Zmud, 1990), de initiatie, gaat namelijk over het actief of passief scannen van problemen of kansen, waarvoor een bepaalde IT oplossing bestaat (zie tabel 6). In dit geval is CDV de (IT) oplossing, waarbij de voordelen van CDV als kans geïdentificeerd dienen te worden.
Daarmee wordt het model van Laukkanen et al. (2017), gecombineerd met het IT implementatieproces (Cooper & Zmud, 1990). Het succesvol doorlopen van de initiatiestap leidt tot het identificeren van bepaalde voordelen als kans. Deze voordelen die CDV biedt, zijn besproken in onderdeel 2.3.1. De volgende stap in het proces is adoptie, waarbij het resultaat de adoptiebeslissing is. Laukkanen et al. (2017) bedoelen met de term adoptie overigens de volledige implementatie, niet het maken van de adoptiebeslissing. Hierdoor kan verwarring ontstaan met wat ik in dit onderzoek bedoel met adoptie. Voordat een organisatie begint aan het implementeren van een oplossing, wordt eerst de adoptiebeslissing gemaakt. Dit komt terug in figuur 4, waar de rode lijn aangeeft dat na de adoptiebeslissing in het IT implementatieproces, de organisatie start met het uitvoeren van implementatieactiviteiten (in figuur 3 adoption actions).



Figuur 4: Model van Laukkanen et al. (2017, p. 57) geprojecteerd op het IT implementatiemodel (Cooper & Zmud, 1990)

Dit onderzoek gaat in op de eerste stappen: initiatie en adoptie. Deze stappen horen bij een organisatie, die CDV nog niet heeft geïmplementeerd. Dit komt terug in figuur 4, links van de rode lijn. De volgende vier stappen, rechts van de rode lijn, leiden tot volledige implementatie. Wanneer echter sprake is van een organisatie, die CDV heeft geïmplementeerd zoals bedoeld door Laukkanen et al. (2017), is niet onderdeel van dit onderzoek. De plaatsing van de stappen rechts van de rode lijn, ten opzichte van het model van Laukkanen et al. (2017), is betekenisloos.

2.3.3. Factoren van invloed op de adoptiebeslissing

Laukkanen et al. (2017) geven aan zich te richten op het maken van de beslissing om CDV te implementeren. Dit doen ze door problemen in kaart te brengen die implementatie in de weg staan. Het werk van Shahin et al. (2017) is zeer vergelijkbaar en gaat eveneens in op problemen bij de implementatie van CDV. Dit heeft geleid tot verschillende uitgebreide overzichten van problemen en hoe deze op te lossen. Beide onderzoeken benadrukken een deel van deze problemen door ze als kritiek te bespreken. Deze kritieke problemen zijn mogelijke factoren van invloed op de adoptiebeslissing, of kunnen er mogelijk een basis voor vormen. Laukkanen et al. (2017) geven bijvoorbeeld aan dat de gerapporteerde kritieke problemen, de grootste impact hadden als obstakel voor de implementatie van CDV. Shahin et al. (2017) rapporteren kritieke factoren, die erg belangrijk zijn bij het succesvol implementeren van CDV.
Deze kritieke problemen zijn niet een-op-een te vergelijken, omdat de onderzoeksresultaten in structuur verschillen. Laukkanen et al. (2017) komen uit op 40 problemen, verdeeld over zeven thema's. Hiervan worden 13 van de 40 problemen als kritiek beschouwd, die verdeeld zijn over vijf van de zeven thema's. Verder zijn 28 causale relaties tussen de problemen gevonden en 29 oplossingen.
Shahin et al. (2017) komen uit op 20 problemen, waarvoor 13 oplossingen zijn gevonden en 30 manieren met bijbehorende tools om tot deze oplossingen te komen. Deze drie onderwerpen zijn opnieuw geanalyseerd door de auteurs en daar zijn zeven kritieke factoren uit gekomen.
In het werk van (Rodríguez et al., 2017) wordt relatief kort ingegaan op de problemen bij de implementatie van CDV, waaruit uiteindelijk vier concrete problemen samengevat kunnen worden.

De (kritieke) problemen van de drie onderzoeken heb ik gecombineerd en geïnterpreteerd, resulterend in vijf determinanten (tabel 8). Wanneer een probleem uit slechts één onderzoek naar voren kwam, heb ik deze buiten beschouwing gelaten en niet meegenomen als determinant. Voor

testen, klant en domein komen de beschrijvingen van de onderzoeken in dusdanige mate overeen, dat de interpretatie ervan evident is. *Transforming towards CD (Rodríguez et al., 2017), human and organizational (Laukkanen et al., 2017), team awareness and transparency, highly skilled and motivated team (Shahin et al., 2017),* gaan in op houding en gedrag van mensen, hoe ze met elkaar omgaan en hoe hun samenwerken georganiseerd wordt. Ik vat dat samen als cultuur. *Build design, system design* (Laukkanen et al., 2017), *Good design principles* (Shahin et al., 2017) gaan in op de vereisten waar een systeem aan zou moeten voldoen en vat ik samen als architectuur. Hoewel het lijkt alsof architectuur een onderdeel is van system design (Laukkanen et al., 2017), blijkt doorgaans dat architectuur juist het overkoepelende onderwerp is (Chen, 2015b; Humble, 2017; Shahin et al., 2019). Wat goed is om te beseffen, is dat Rodríguez et al. (2017) architectuur dan niet expliciet noemen als probleem, maar wel als component van CDV (zie tabel 3). Uit de beschrijving blijkt dat het om een kritieke component gaat, wat bijdraagt aan de overtuiging om architectuur als determinant op te nemen.

Geïnterpreteerde determinant	**Beschreven problemen (Rodríguez et al., 2017)**	**Kritieke problemen (Laukkanen et al., 2017)**	**Kritieke factoren (Shahin et al., 2017)**
Testen	*Increased efforts in testing/QA activities*	*Testing (Ambiguous test result, Time-consuming testing, Multi-platform testing, Flaky tests, Problematic deployment)*	*Testing (effort and time)*
Cultuur	*Transforming towards CD*	*Human and organizational (Organizational structure, Lack of motivation)*	*Team awereness and transparency*
			Highly skilled and motivated team
Klant	*Customer unwillingness*		*Customer*
Domein	*Domain*		*Application domain*
Architectuur		*Build design (inflexible build)*	*Good design principles*
		System design (unsuitable architecture, system modularization, internal dependencies)	
			Appropriate infrastructure
		Integration (Broken build, slow integration approval)	

Tabel 8: Combinatie en interpretatie van problemen tot determinanten

De vijf gevonden determinanten worden hieronder verder uiteengezet. Tot slot zijn de determinanten samengevat in tabel 10.

Klant

Een hoge frequentie van nieuwe versies van een softwareproduct, betekent ook dat klanten steeds te maken krijgen met die nieuwe versies. Mogelijk moet een nieuwe versie lokaal geïnstalleerd worden en dienen klanten de nieuwe functionaliteiten of wijzigingen in de nieuwe versie onder de

knie te krijgen. Het blijkt dat niet alle klanten dit altijd willen, met name omdat onstabiele software als ongewenst wordt ervaren (Rodríguez et al., 2017; Shahin et al., 2017). Dit geldt voor CDP vanzelfsprekend sterker dan voor CDV en hangt samen met de manier waarop de software uitgerold wordt. In een bedrijfsomgeving met een complexe infrastructuur levert het uitrollen van een nieuwe softwareversie van een administratiesysteem een veel grotere uitdaging op dan wanneer een nieuwe versie van een webapplicatie wordt aangeboden middels een browser.
In bepaalde situaties vinden klanten het uitstekend wanneer ze iedere, bijvoorbeeld, 4 maanden een nieuwe versie van de software krijgen aangeleverd (Leppanen et al., 2015)
Het kan daarom een overweging zijn om strikt voor de implementatie te gaan van CDV, zonder CDP. Wanneer software in continue staat van paraatheid (uit te rollen naar productie) wordt gehouden, hoeft een klant daar geen last van te hebben. De factor klant heeft dus met name invloed op de beslissing in welke mate CDV geïmplementeerd kan worden, inclusief CDP of niet.

Domein
Rodríguez et al. (2017) geven aan dat CDV binnen het domein van geïntegreerde systemen (embedded systems) hevig bemoeilijkt wordt. Dit wordt gerelateerd aan de architectuur die gepaard gaat met dit domein. Dit wordt bevestigd door Shahin et al. (2017), die aangeven dat het domein bepalend is voor de keuze CDV of CDP. CDV zou altijd moeten kunnen, maar de mogelijkheid voor CDP is afhankelijk van het software domein, in lijn met bovenstaande tekst over de klant. Shahin et al. (2017) relateren de klant en het domein dan ook aan elkaar en doen een pleidooi om het applicatiedomein en de beperkingen van de klant grondig te analyseren, alvorens CDP te implementeren. Zoals eerder in dit onderzoek opgemerkt wordt CDP niet uitgesloten door CDV. De determinanten klant en domein geven daar meer duiding aan. De keuze CDV of CDP hoeft dus niet binair gemaakt te worden. Het onderzoek van Leppanen et al. (2015) laat 15 organisaties zien die allen beschikken over zogenaamde CDP mogelijkheden, maar die geen van allen CDP volledig tot en met geautomatiseerde uitrol naar productie hadden gerealiseerd. De auteurs geven verschillende niveaus, voor het met CDV/CDP na te streven doel, waarbij duidelijk wordt dat het domein hierop van invloed is. De voordelen die CDV biedt kunnen ook gezien worden als het doel dat nagestreefd wordt en dit is bepalend voor de mate waarin CDV geïmplementeerd dient te worden.

Testen
Continu testen vormt de kern van CDV en kan alleen gerealiseerd worden middels automatisering (Humble & Farley, 2010; Rodríguez et al., 2017). Overgebleven handmatige tests, langdurende tests en testcases die met hoge frequentie falen, zitten de te behalen voordelen van CDV in de weg (Shahin et al., 2017). De testproblemen die Laukkanen et al. (2017) geven, gaan gepaard met oplossingen, zijn gerelateerd aan een ongeschikte architectuur en gelijk aan eerder genoemde (*time-consuming testing)* en voegen daarom niet veel toe. Uit nagenoeg alle bronnen blijkt expliciet dat testen een kritieke component is binnen CDV en er bestaat geen relevante discussie over (e.g.: Akerele, 2018; Chen, 2017; Claps et al., 2015; Humble & Farley, 2010; Laukkanen et al., 2017; Neely & Stolt, 2013) Samenvattend stellen Rodríguez et al. (2017) dat de inspanning op testen erg toeneemt. Vanaf het begin van CDV wordt ook duidelijk aangegeven dat het ontwikkelen van een teststrategie centraal staat (Humble & Farley, 2010).

Architectuur
Laukkanen et al. (2017) merken in hun onderzoek op dat een geschikte architectuur de meest kritieke enabler is voor CDV. Rodríguez et al. (2017) geven aan dat CDV een softwarearchitectuur vereist, die continue evolutie en aanpassing van het product naar aanleiding van veranderende

requirements, mogelijk maakt. Shahin et al. (2017) leggen uit dat architectuur een van de belangrijkste factoren is die implementatie van continuous practices mogelijk maken. De architectuur staat aan de basis van hoe software wordt ontwikkeld, geïntegreerd, getest en uitgerold. Daarom is de fase waarin de softwarerarchitectuur wordt ontwikkeld erg belangrijk (Shahin et al., 2017). Bij CDV wordt immers gestreefd naar het zoveel mogelijk automatiseren van deze activiteiten (Humble & Farley, 2010), dus de architectuur moet het geautomatiseerd uitvoeren van deze activiteiten mogelijk maken. Het meenemen van dit feit in een greenfield project is logischerwijs van een heel andere orde, dan een bestaande softwarearchitectuur te herontwikkelen. Een voorbeeld waarbij CDV met moeite werd geïmplementeerd kwam mede doordat het vier jaar duurde voordat er een nieuwe architectuur ontwikkeld was (Humble, 2017). Het is duidelijk dat CDV implementeren voor een legacy oplossing veel uitdagender is dan voor een greenfield project, of zelfs niet mogelijk is (Chen, 2017). Organisaties met oude, monolithische systemen zouden deze systemen moeten herarchitecteren[2] om CDV gemakkelijker te kunnen implementeren (Shahin et al., 2019). Monolithische (legacy) systemen sluiten CDV niet uit, maar het is veel moeilijker CDV te implementeren. De redenen hiervoor zijn onder andere dat het geautomatiseerd testen en uitrollen van software maar beperkt lukt en dat feedback langzaam en inconsistent is. Dit heeft tot gevolg dat software niet snel en frequent uitgerold kan worden (Shahin et al., 2019). Uit de componenten van CDV komt duidelijk naar voren dat deze dynamiek de kern van CDV vormt. Ondanks dat uit empirisch onderzoek blijkt (Shahin et al., 2019) dat organisaties aangeven dat zij CDV kunnen implementeren in dergelijke omgeving, kun je je daarom afvragen in hoeverre echt sprake kan zijn van CDV. De componenten kunnen immers niet goed toegepast worden en de voordelen worden niet behaald.

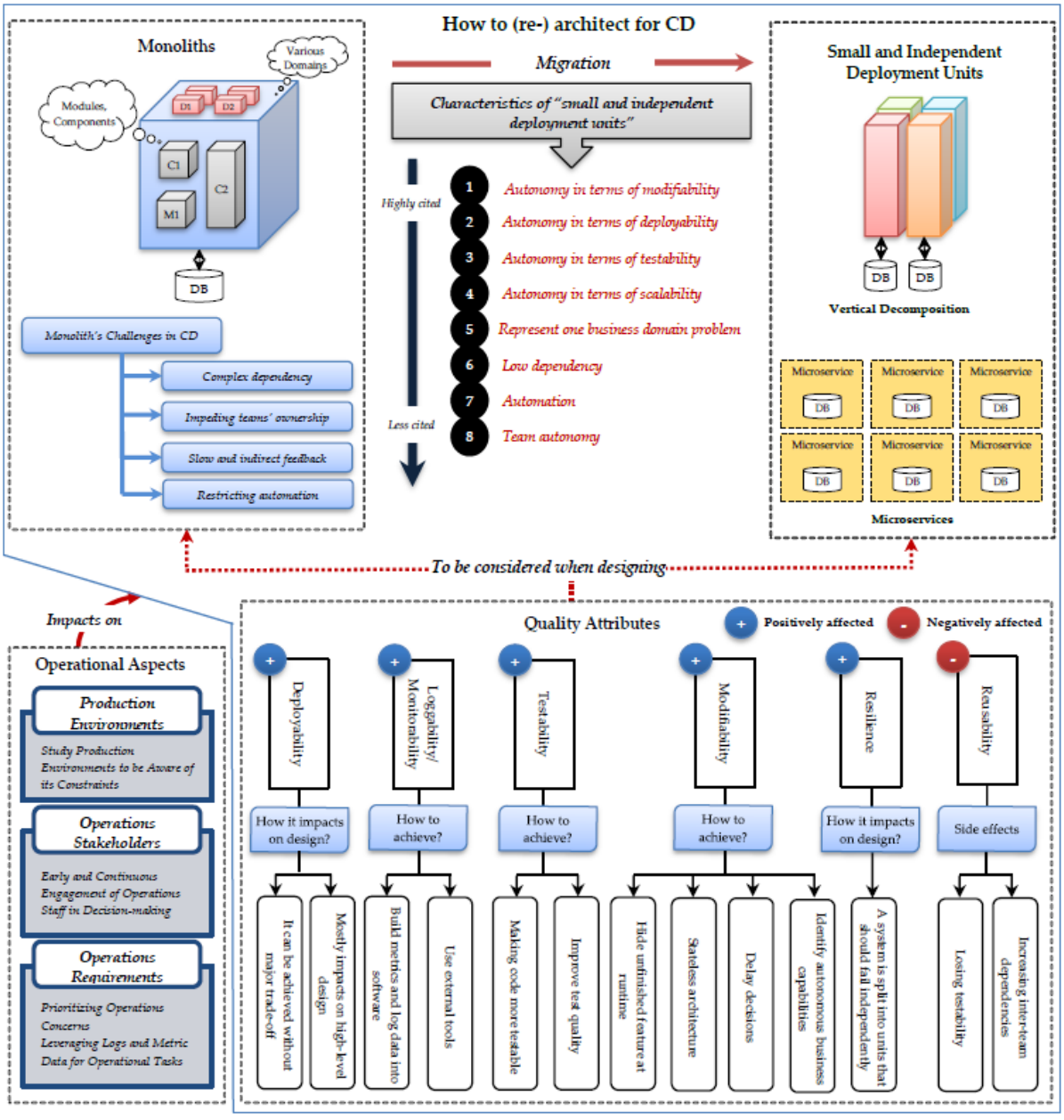
Chen (2015b)	**Shahin et al. (2019)**	**Humble (2017)**
Deployability	Deployability	Deployability
Security		
Loggability	Loggability/monitorability	
Monitorability		
Modifiability	Modifiability	
Testability	Testability	Testability
	Resilience	
	Reusability	

Tabel 9: Aspecten binnen een voor CDV geschikte architectuur

Er is meer bekend over de aspecten binnen de softwarearchitectuur in relatie tot CDV. Allereerst benoemt Chen (2015b) zes aspecten die geborgd zouden moeten zijn (tabel 9).
Vervolgens leggen Shahin et al. (2019) uit dat de essentie van de CDV softwarearchitectuur gaat om het principe van kleine eenheden, die onafhankelijk van elkaar uitgerold kunnen worden. Teneinde dit te bereiken noemen zij eveneens zes aspecten die binnen de softwarearchitectuur van belang zijn. De aspecten van de drie publicaties overlappen elkaar deels (zie tabel 9). Shahin et al. (2019) gaan dieper in op deze aspecten dan Chen (2015b) en leggen uit hoe de aspecten de architectuur beïnvloeden en hoe dat te bereiken. Voor security wordt niets gevonden door Shahin et al. (2019) en de onderbouwing van de security impact op architectuur van Chen (2015b) is mager.
In het werk van Humble (2017) is eveneens te lezen dat de CDV architectuur geschikt moet zijn voor onafhankelijk uit te rollen componenten. Hij noemt echter alleen de aspecten met betrekking tot testen en uitrollen (deploy). Deze aspecten zouden echter zijn geïntroduceerd door het fenomeen CDV (Humble, 2017), wat kan betekenen dat de andere aspecten doorgaans al als belangrijk werden

[2] Architecteren is geen woord, maar zou het werkwoord voor architectuur kunnen zijn. In het Engels wordt wel *architecting* als werkwoord gebruikt.

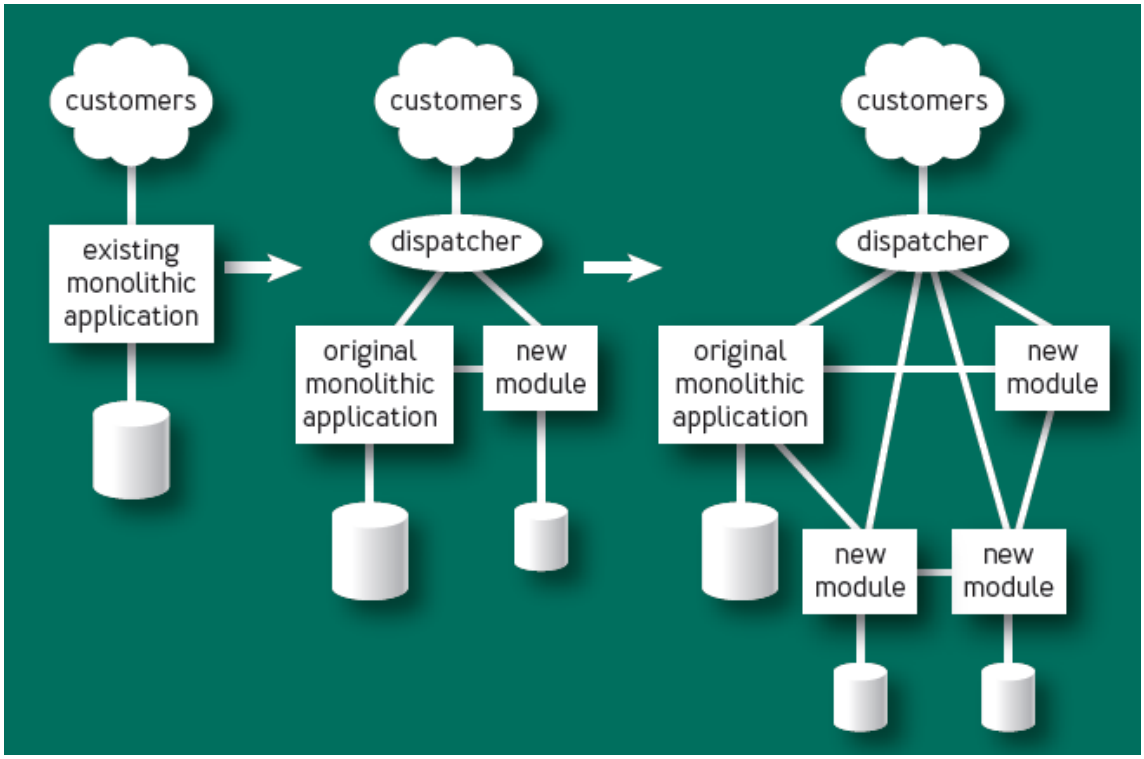
gezien binnen softwarearchitectuur. Er wordt echter ook aangegeven dat al deze aspecten altijd al van belang waren, maar dat CDV de prioriteit ervan dusdanig verhoogt dat ze van kritieke aard zijn (Shahin et al., 2019).
De verschillen in architectuuraspecten tussen Shahin et al. (2019) en Chen (2015b) zijn beperkt, waarbij Shahin et al. (2019) niet worden tegengesproken. Dit gebeurt ook niet door Humble (2017). Derhalve neem ik de aspecten van Shahin et al. (2019) over, ook omdat deze veel uitgebreider zijn uitgewerkt. Zie ook figuur 5.



Figuur 5: Conceptueel framework met bevindingen over hoe te komen tot een CDV architectuur (Shahin et al., 2019, p. 11)

Tevens wordt duidelijk gemaakt hoe een organisatie van een oude situatie met een monolithisch systeem, kan ontwikkelen naar een CDV situatie (Humble, 2017; Shahin et al., 2019). Dit komt terug in de figuren 5 en 6. Hierin wordt duidelijk waarom architectuur een kritiek onderwerp is bij het maken van de adoptiebeslissing. Een systeem dat niet over de architectuur en eigenschappen beschikt waarmee CDV mogelijk is, kan daarvoor geschikt gemaakt worden door het systeem opnieuw te ontwikkelen, waarbij het opgeknipt wordt in kleine, onafhankelijke modules. Dit

betekent in feite dat uiteindelijk het gehele systeem opnieuw gebouwd is binnen een voor CDV geschikte architectuur.



Figuur 6: Ontwikkeling van monolithisch systeem naar CDV architectuur (Humble, 2017, p. 17)

Cultuur

Rodríguez et al. (2017) bespreken het component 'organisatorische factoren' en benoemen daarbij dat bedrijfsfuncties geïntegreerd moeten zijn in het softwareontwikkelteam (1), dat sprake moet zijn transparantie (2) en dat een innovatieve en experimentele organisatiecultuur aanwezig moet zijn (3). Hoewel dat niet direct uit de tekst blijkt, lijkt de laatste kritiek. Wanneer dergelijke cultuur aanwezig is, zal het geen probleem zijn om tot integratie en transparantie te komen. In verschillende andere literatuur komt ook terug dat cultuur een kritieke succesfactor is bij het bereiken van CDV/CDP (Claps et al., 2015). Daarbij wordt benoemd dat meer nodig is dan samenwerking van ontwikkelaars, wat overeenkomt met het de benodigde integratie van bedrijfsfuncties (Rodríguez et al., 2017). CDV gaat in de basis over continu verbeteren en om dit te bereiken zal een toereikende mate van bevoegdheid aan uitvoerende teams gegeven moeten worden (Humble, 2017). Dit is in lijn met de principes van Lean softwareontwikkeling (empower the team) (Claps et al., 2015; Rodríguez et al., 2017). Humble (2017) legt vervolgens uit wat cultuur is en welke cultuur nodig is om CDV tot uiting te laten komen. Daarvoor gebruikt hij het model van Westrum, bestaande uit drie verschillende culturen (figuur 7). Dit model is gebruikt om de impact van cultuur op digitale systemen te onderzoeken. Westrum stelt daarbij dat het belangrijk is om een cultuur te creëren waarin nieuwe ideeën omarmd worden, mensen van over de gehele organisatie samenwerken aan gemeenschappelijke doelen, mensen getraind worden in het rapporteren van slecht nieuws zodat daaraan gewerkt kan worden en fouten en ongelukken te behandelen als kansen om van te leren, in plaats van zoeken naar de schuldigen ervan (Humble, 2017).

PATHOLOGICAL (POWER-ORIENTED)	BUREAUCRATIC (RULE-ORIENTED)	GENERATIVE (PERFORMANCE-ORIENTED)
low cooperation	modest cooperation	high cooperation
messengers shot	messengers neglected	messengers trained
responsibilities shirked	narrow responsibilities	risks are shared
bridging discouraged	bridging tolerated	bridging encouraged
failure leads to scapegoating	failure leads to justice	failure leads to enquiry
novelty crushed	novelty leads to problems	novelty implemented

Figuur 7: Drie culturen model van Westrum (Humble, 2017, p. 14)

Deze cultuur staat aan de basis voor het samenwerken van ontwikkelaars en beheerders in één team (Humble, 2017), wat overeenkomt met de benodigde integratie van bedrijfsfuncties (Rodríguez et al., 2017) en versterkt het idee dat een innovatieve en experimentele cultuur hieraan voorafgaat.

Determinant	Beschrijving
Klant	Een hoge frequentie van nieuwe softwareversies is voor de klant niet altijd gewenst. De aard van het (software)product en bijbehorende doelgroep dienen zorgvuldig in kaart te worden gebracht, alvorens CDV geïmplementeerd wordt en bij het bepalen in welke mate CDV geïmplementeerd wordt.
Domein	Het applicatiedomein heeft invloed op de manier waarop de software uitgerold kan worden naar een volgende omgeving en uiteindelijk de klant. Dit kan beperkingen geven aan de mate waarin CDV geïmplementeerd kan worden en de voordelen tot uiting kunnen komen. Derhalve dient het applicatiedomein grondig geanalyseerd te worden, alvorens CDV geïmplementeerd wordt.
Architectuur	Een geschikte architectuur is een zeer kritieke factor bij het realiseren van CDV. De software dient geautomatiseerd getest en uitgerold (deploy) te worden en dat kan alleen wanneer de architectuur van de software zich daarvoor leent. Een voor CDV geschikte architectuur wordt gevormd door de volgende aspecten: • Deployability - uitrolbaarheid: software dient te bestaan uit componenten die (betrouwbaar en gemakkelijk) los van elkaar uit te rollen zijn. • Modifiability - aanpasbaarheid: software dient gemakkelijk en vaak aangepast te kunnen worden, zonder dat dit een negatief effect heeft op ander software componenten. • Loggability and monitorability - vermogen om logging en monitoring toe te passen: bij het continu ontwikkelen, testen en uitrollen van software gaan dingen fout. Deze fouten dienen snel opgespoord en hersteld te worden, wat veel logging vereist. Deze logdata dient op een slimme manier gemonitord te worden. • Testability – testbaarheid: software dient snel en geautomatiseerd getest te kunnen worden. • Resilience – veerkracht: ga er vanuit dat componenten weleens kapot gaan. Het omvallen van een bepaald softwarecomponent dient dan minimale gevolgen te hebben. • Reusability – herbruikbaarheid: inzetten op herbruikbaarheid heeft een negatief effect op CDV, omdat onderlinge afhankelijkheden tussen componenten worden gecreëerd. Bij een ongeschikte architectuur, vaak te vinden bij oudere, monolithische software, zal sprake zijn van een langdurig en kostbaar verandertraject om de software -onder architectuur- opnieuw te bouwen in componenten.
Cultuur	CDV is een fenomeen in de lijn van Agile en Lean softwaredevelopment. Het vereist een cultuur van continu verbeteren, waarin nieuwe ideeën enthousiast worden omarmd, fouten worden gezien als kansen om van te leren, professionals ruimte en vertrouwen krijgen zodat zij sterk gemotiveerd blijven. Dit leidt tot het bewustzijn en de transparantie die vereist zijn om CDV tot een succes te maken.
Testen	De testinspanning van de organisatie zal zeer sterk toenemen en dient niet onderschat te worden. Handmatige tests en langdurige tests vormen een risico op het succes van CDV en dienen tot een minimum beperkt te worden.

Tabel 10: Vastgestelde determinanten

2.4. Doel van het vervolgonderzoek

Uit het literatuuronderzoek blijken tien componenten en zeven voordelen die gezamenlijk CDV definiëren. Daarnaast is een implementatieproces bepaald, waarbij voor dit onderzoek de nadruk wordt gelegd op het doorlopen van de initiatiefase en adoptiefase, resulterend in, respectievelijk het identificeren van een kans of probleem en in een gemaakte adoptiebeslissing. Tot slot zijn vijf (mogelijke) determinanten geïdentificeerd, waarvan verondersteld wordt dat ze zeer nuttig zijn, wanneer ze betrokken worden bij het maken van de adoptiebeslissing.
In het vervolgonderzoek zijn deze conclusies getoetst aan de empirie. De literatuur gaat ruimschoots in op de componenten en voordelen van CDV. Het is interessant om te onderzoeken of organisaties ze inderdaad identificeren als kans, resulterend uit de initiatiefase. De hoofdbijdrage van het empirisch onderzoek komt uit de bevestiging en/of aanpassing van de geïdentificeerde determinanten voor de adoptiebeslissing. Bij de identificatie van deze determinanten is meer ruimte geweest voor eigen interpretatie, waardoor toetsing aan de empirie essentieel is.

3. Methodologie

Dit hoofdstuk gaat in op wat het vervolgonderzoek inhoudt en hoe het zal worden uitgevoerd.

3.1. Conceptueel ontwerp: keuze van onderzoeksmethode

Het vervolgonderzoek betreft empirisch onderzoek, waarin het verrijken van het theoretisch perspectief centraal staat. Van de bestaande theoretische inzichten, gecombineerd in het voorgaande literatuuronderzoek, zal de houdbaarheid getoetst worden in de praktijk. Mogelijk leidt dit tot het bijstellen of verfijnen van bestaande inzichten. Er is sprake van een inductieve beredenering in het onderzoek, omdat getracht wordt vanuit empirische waarneming, tot generalisatie te komen. Een overweging hierbij is dat het expliciet identificeren van determinanten, die een rol spelen bij het maken van de adoptiebeslissing, voor zover bekend, nog nooit is gedaan. Hoewel een beeld bestaat van CDV als fenomeen, kan het maken van de adoptiebeslissing mogelijk als een fenomeen binnen CDV worden gezien. Derhalve zal kwalitatief onderzoek worden uitgevoerd, wat in lijn is met theorieontwikkeling en een inductieve manier van beredeneren (Saunders, Lewis, & Thornhill, 2016). Deze benadering komt eveneens voort uit de onderzoeksvragen die zijn beschreven in de introductie. Bij deze onderzoeksvragen past een exploratief onderzoek, we willen beter begrijpen hoe organisaties de adoptiebeslissing voor CDV nemen. Daarbij is het belangrijk te beseffen dat de in het literatuuronderzoek geïnterpreteerde determinanten, mogelijk veranderd of verrijkt kunnen worden naar aanleiding van het empirisch onderzoek. Dit betekent dat de onderzoeker open moet staan voor nieuwe inzichten, die alleen naar voren kunnen komen bij diepgaand onderzoek, waarbij organisaties hun ervaringen naar voren kunnen brengen. Om die reden is het afnemen van interviews het meest geschikte middel. Met behulp van een interview is een onderzoeker in staat om valide en betrouwbare data te verkrijgen, die kunnen helpen bij het beantwoorden van de onderzoeksvragen (Saunders et al., 2016).

3.2. Technisch ontwerp: uitwerking van de methode

Het onderzoek gaat in op het maken van de adoptiebeslissing (al dan niet over te gaan tot de implementatie van CDV). Organisaties die bezig zijn met het maken van deze adoptiebeslissing, of dit reeds gedaan hebben, komen dus in aanmerking om onderdeel te zijn van het onderzoek. Organisaties die nog aan het ontdekken zijn wat CDV precies is en of het mogelijk een oplossing voor hun probleem kan zijn, maar nog niet bezig zijn met de adoptiefase zoals bedoeld in het IT-implementatie proces (Cooper & Zmud, 1990), zijn ongeschikt om mee te nemen in het onderzoek. In het zoeken naar geschikte kandidaten is hier rekening mee gehouden.

De kern van het onderzoek zal bestaan uit interviews. Een ongestructureerd of semigestructureerd interview past het beste bij exploratief onderzoek (Saunders et al., 2016). Daarbij zal het onderzoek zich moeten richten op medewerkers binnen organisaties, die de overwegingen maken die komen kijken bij het maken van de adoptiebeslissing. Dit zijn medewerkers met (gedeeltelijke) budgetverantwoordelijk en beslissingsbevoegdheid, of met kennis en expertise die daarop van invloed zijn. Voorbeelden hiervan zijn managers, product-owners en architecten. Het is gebleken dat dergelijke medewerkers liever meewerken aan interviews dan aan surveys, omdat een interview de mogelijkheid biedt in te gaan op hoe bepaalde informatie geïnterpreteerd te worden (Saunders et al., 2016). Dit is in lijn met de eerder vastgestelde, kwalitatieve aanpak.

De onderzoeker is reeds tien jaar werkzaam op het grensvlak van business en IT, in diverse functies en organisaties, waarbij het adequaat gebruiken en doorontwikkelen van software centraal staat. Hierbij is een netwerk ontstaan, waarin naar verwachting voldoende geschikte kandidaten gevonden kunnen worden om dit onderzoek mee uit te voeren.
In het literatuuronderzoek is naar voren gekomen dat het resultaat van de eerste fase (initiatie) van het IT-implementatieproces, organisaties geïdentificeerde kansen (problemen) geeft, waar de voordelen van CDV op aansluiten (als oplossing of te benutte kans). Alvorens organisaties de adoptiebeslissing voor CDV maken, is te verwachten dat zij de voordelen die CDV biedt als kans of oplossing, geïdentificeerd hebben en dit meenemen in hun overwegingen. Tevens is te verwachten dat organisaties weten wat CDV is, wat in het literatuuronderzoek terugkomt in de componenten van CDV.
Daarnaast is, op basis van het literatuuronderzoek, te verwachten dat de geïnterpreteerde determinanten terugkomen in de overwegingen van organisaties en een rol spelen bij het maken van de adoptiebeslissing. Hoewel de (tweede) onderzoeksvraag hoofdzakelijk gaat over de rol die de determinanten spelen bij het maken van de adoptiebeslissing, zullen de componenten van CDV en voordelen die CDV biedt, naar verwachting dus ook onderdeel zijn van de overwegingen en daarom ook worden meegenomen in het onderzoek. Om deze theoretische kennis uit het literatuuronderzoek optimaal te betrekken in het empirisch onderzoek, wordt een semigestructureerd interview afgenomen (Saunders et al., 2016). Hierbij zullen de overzichten van tabellen 3, 6, 7 en 10 (zie hoofdstuk 2) worden verwerkt in overzichten die ter ondersteuning gebruikt worden tijdens het interview. Zie hiervoor bijlage 3. Daarnaast heeft de onderzoeker een interviewschema opgesteld, waarin de thema's van het interview en de belangrijkste vragen terugkomen, om richting te geven aan het interview en de onderzoeker in staat te stellen deze thema's en vragen in ieder interview te laten terugkomen. Zie hiervoor bijlage 4. Hierbij zijn vooraf gesloten vragen opgesteld, hoofdzakelijk als introductie, en open vragen, bedoeld om kandidaten uitgebreid op een onderwerp in te laten gaan. Naast deze twee type vragen beschrijven Saunders et al. (2016) ook de onderzoekende vraag, die tijdens het interview terug zullen komen wanneer op een specifiek onderwerp doorgevraagd wordt. Enkele van deze vragen zijn al opgesteld, maar afhankelijk van het verloop van het interview hoeven ze mogelijk niet gesteld te worden.
Na introductie wordt een belangrijk onderdeel van het interview gevormd door het thema adoptiebeslissing (zie bijlage 4). Hierbij wordt de open vraag gesteld hoe organisaties tot het besluit zijn gekomen de CDV werkwijze te implementeren. Vervolgens wordt meer structuur aangebracht in het interview door de voordelen en componenten van CDV te bespreken en te betrekken op het maken van de adoptiebeslissing. Tot slot komen de geïnterpreteerde determinanten aan bod, waarbij wordt getoetst of ze een rol speelden bij het maken van de adoptiebeslissing en of dat zo had kunnen zijn, mocht de organisatie bewuster zijn geweest van deze determinanten. Tevens wordt gevraagd naar eventuele overige determinanten.
De eerste open vraag kan al een gewenst resultaat van het interview opleveren. De thema's die volgen zijn vooral bedoeld om deze onderwerpen in ieder interview aan bod te laten komen.

3.3. Gegevensanalyse

De interviews zullen worden opgenomen, waarna er transcripten van worden opgesteld. Vervolgens kan de data gecodeerd worden, zodat thema's en relaties gemakkelijker geïdentificeerd kunnen worden. Wanneer documenten uit de organisatie toegevoegd kunnen worden aan de data, dan volgen deze documenten het proces vanaf coderen. De stappen in het analyseren van de kwalitatieve data komen daarmee overeen met de thematische analyse (Saunders et al., 2016). Dit is

een generieke methode en daarmee geschikt voor dit onderzoek. Een nadeel is mogelijk dat het geen specifiek theorie toetsende methode is.

3.4. Reflectie t.a.v. validiteit, betrouwbaarheid en ethische aspecten

Alleen medewerkers die betrokken zijn geweest bij het maken van de adoptiebeslissing komen in aanmerking om mee te doen aan het onderzoek. Het uitgangspunt van het onderzoek is dat alleen dergelijke medewerkers kunnen bijdragen aan een valide onderzoeksresultaat. Het interviewschema is met zorg opgesteld en geeft richting aan een interview met zoveel mogelijk open vragen, ter bevordering van de validiteit. De thema's in het interviewschema zorgen ervoor dat dezelfde onderwerpen in ieder interview terugkomen, wat de betrouwbaarheid van het onderzoek ten goede komt. Door kritisch door te vragen op de rol die de determinanten speelden bij het maken van de adoptiebeslissing, zal de validiteit van de antwoorden zoveel mogelijk worden bevestigd. Voor de vakspecifieke termen zal een definitie paraat zijn, om misinterpretatie te voorkomen. Om de betrouwbaarheid te verhogen, wordt gestreefd naar twee interviews per organisatie. Het totaal aantal interviews dat afgenomen wordt, heeft niet zoveel invloed op de validiteit als het doorgronden van de verzamelde data (Saunders et al., 2016). Daarnaast is het bijzonder lastig om specifiek aan te geven hoeveel interviews afgenomen moeten worden om tot een betrouwbaar resultaat te komen en er zijn dan ook geen strikte regels (Saunders et al., 2016) Toch raden Saunders et al. (2016) aan om, voor een onderzoek in het algemeen, minimaal 5-25 interviews aan te houden. Voor dit onderzoek streef ik dan ook naar een minimum van vijf interviews.
Na de interviews zullen de transcripten gedeeld worden met de geïnterviewden, om interpretatieproblemen te minimaliseren. Dit past tevens bij een ethisch verantwoorde omgang met de kandidaten (Saunders et al., 2016). De verwachting is dat het onderwerp van het onderzoek niet van dusdanig gevoelige aard is, dat een betekenisvolle kans op sociaal-wenselijke antwoorden bestaat.
Ten aanzien van de ethische aspecten kan verder worden opgemerkt dat deelname aan het onderzoek vrijwillig is. Deelnemers hoogstens met generieke functie worden genoemd en verder anoniem blijven, evenals hun organisatie. Zij zullen vooraf toestemming moeten geven voor de gespreksopname en de gespreksopname zal nadien gewist worden. Deze maatregelen moeten leiden tot een verantwoord onderzoek.

4. Resultaten

In dit hoofdstuk wordt uitgelegd hoe het empirisch onderzoek, beschreven in hoofdstuk 3, is uitgevoerd. Op welke manier data verzameld is en hoe deze data zijn verwerkt.

4.1. Data verzamelen en verkennen

Voor het onderzoek zijn in totaal zes interviews gehouden met professionals. Dit waren twee (Enterprise) Architecten, twee (senior) Product Owners en twee Managers (een lijnmanager en een programmamanager). Vijf deelnemers zijn enkele tot vele jaren werkzaam in hun functie bij de betreffende organisatie. Een deelnemer is niet meer werkzaam bij de betreffende organisatie en deelt ervaringen van enige jaren geleden. De deelnemers zijn dusdanig betrokken (geweest) bij het proces van softwareontwikkeling dat hun kennis en ervaring als zeer relevant wordt beschouwd voor dit onderzoek.
Dit is geheel volgens plan verlopen. De inschatting die vooraf is gemaakt over geschikte deelnemers en de uitnodigingstekst om ook de deelnemers bewust te maken van hun geschiktheid, bleken correct en effectief. Het interviewschema is niet aangepast. De interviews zijn gehouden in de periode maart-december 2020. De interviews zijn op afstand afgenomen met behulp van videoconferentie. Gezien de getroffen coronamaatregelen was dit de enige voor de hand liggende manier en dit heeft niet tot enige discussie of overweging geleid.

In eerste instantie is een volledig transcript gemaakt van het interview. Dit transcript bleek niet effectief bij het coderen van de data. Daarop is een transcriptsamenvatting gemaakt. Deze transcriptsamenvatting is gebaseerd op het interview en niet op het transcript ervan. Het voordeel hiervan is dat de toon en non-verbale communicatie zodoende direct in de transcriptsamenvatting verwerkt kunnen worden. Het (steeds opnieuw) beluisteren van de opname is daarmee een belangrijke manier geweest om bekend te raken met de data. De transcriptsamenvattingen leggen de basis voor het coderen en verdere verwerking en zorgen voor verifieerbaarheid. De transcriptsamenvattingen zijn ondergebracht in bijlage 5. Voor slechts één interview is een volledig transcript opgesteld. Deze is niet gebruikt en is ook niet opgenomen in de bijlagen.

Ten opzichte van het plan zijn twee afwijkingen ontstaan in de uitvoering van het onderzoek. De deelnemers is niet gevraagd om de uitwerking te bevestigen. Hiermee zal rekening gehouden moeten worden bij de beoordeling van de betrouwbaarheid. Het idee om per organisatie te streven naar twee deelnemers is losgelaten. De achterliggende scope voor de interviews zou per deelnemer kunnen verschillen. Twee Product Owners binnen dezelfde organisatie, kunnen bijvoorbeeld volstrekt verschillende aandachtsgebieden hebben. De betrouwbaarheid van het interview zou hiermee niet substantieel hoger worden. Daarnaast bleek het ook in de praktijk lastig om twee geschikte deelnemers te vinden bij dezelfde organisatie.

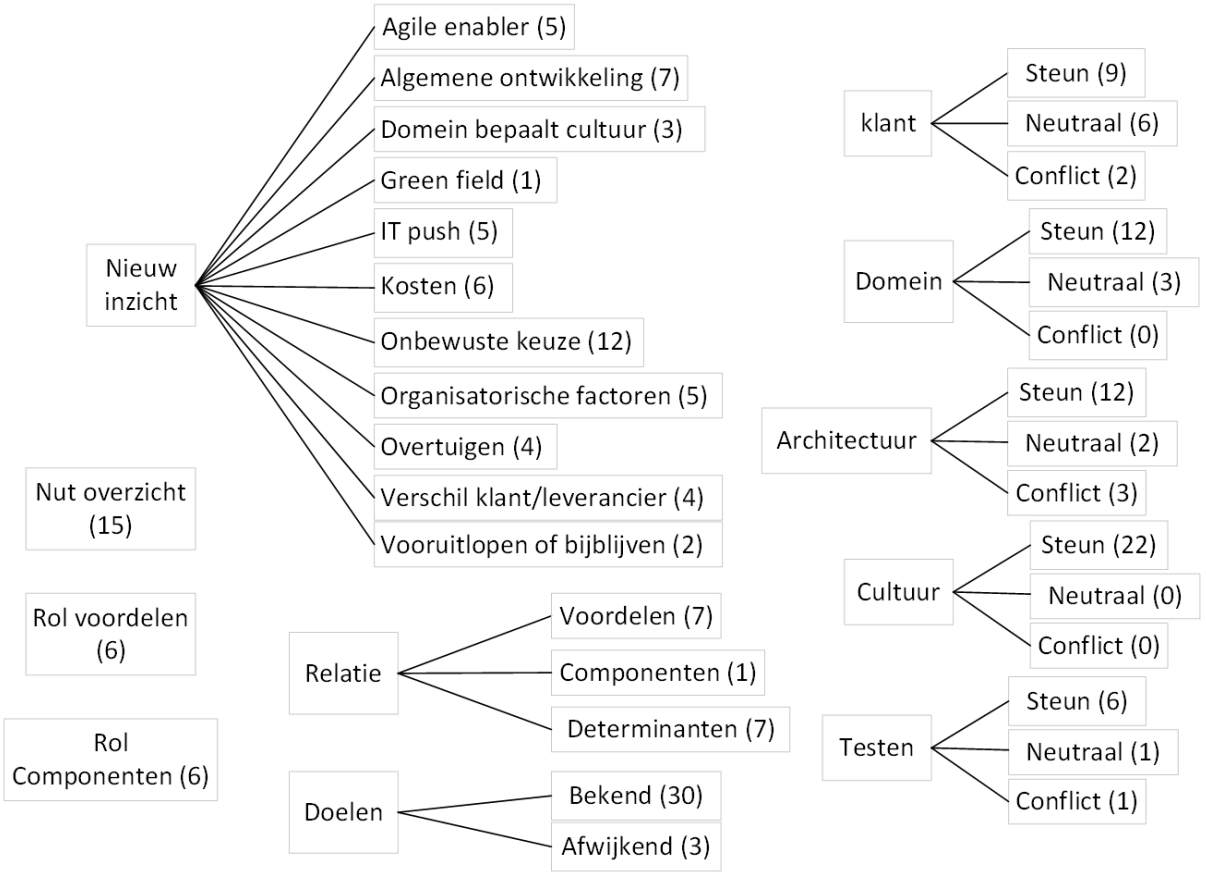
Bij de interviews kwam een structuur naar voren die reeds te ontdekken is in het interviewschema. Allereerst is getracht de situatie en enige achtergrond te schetsen. Bij de inleidende vragen over de status van CDV en de releasefrequentie kwamen bij verschillende interviews al uitgebreide antwoorden naar voren. In andere gevallen werd hier kort en eenduidig op geantwoord. Voor een deel van de interviews gold dat de vragen over de doelen die een organisatie nastreeft met CDV en de voordelen van CDV in elkaar verweven raakten en dat hier geen strikt onderscheid in wordt gemaakt. In andere gevallen komt dat onderscheid meer naar voren.

De volgende structuur is steeds terug te vinden in de transcriptsamenvattingen:

1. Inleiding en introductie
2. (eventueel) Status CDV
3. Adoptiefase CDV
4. Doelen/voordelen CDV
5. (eventueel) Overzicht voordelen CDV
6. Determinanten CDV

4.2. Coderen en verfijnen

Zoals uitgelegd in hoofdstuk 3 is sprake van inductief onderzoek. Het ligt daarbij voor de hand om codes uit de data te onttrekken ('in vivo') (Saunders et al., 2016). Om hierin te versnellen helpt uiteraard de onderzoeksvraag. In het algemeen wordt gezocht naar hoe organisaties de adoptiebeslissing maken. Meer specifiek wordt gezocht naar determinanten, waarvan het literatuuronderzoek er reeds vijf heeft opgeleverd. Deze vijf determinanten worden daarom als codes gebruikt. Daarmee wordt niet zuiver 'in vivo' gecodeerd, maar worden codes gebruikt die de onderzoeker heeft bedacht. Uitspraken die niet direct of gemakkelijk aan een van deze determinanten gekoppeld kunnen worden, vormen een nieuw inzicht. Primair ontstaan daarmee de volgende codes: Nieuw inzicht, Klant, Domein, Architectuur, Cultuur, Testen.
Bij het analyseren van de data blijkt dat er veelvuldig gesproken wordt over de doelen die men nastreeft met CDV. Verder viel op dat relaties vaak ter sprake kwamen. Gedurende het interview werd expliciet gevraagd naar de bruikbaarheid van de gedeelde overzichten en naar de rol die de voordelen en componenten hebben gespeeld. Deze constateringen leiden tot de codes: Doelen, Relaties, Nut overzicht, Rol voordelen, Rol componenten.
Na verfijnen en bijstellen, waarbij verschillende iteraties zijn doorlopen, zijn de nieuwe inzichten op een dieper niveau uitgewerkt. Bij de doelen bleken de uitspraken weinig onderscheidend. Derhalve is gekeken of een beschreven doel past binnen de gevonden voordelen van CDV uit het literatuuronderzoek, of dat het daarvan afwijkt. Voor de determinanten is onderscheid gemaakt in uitspraken die de determinanten ondersteunen of ermee conflicteren. Uitspraken konden echter ook neutraal zijn. De relaties zijn verfijnd in componenten, voordelen en determinanten. Het nut van de overzichten en de rol van de voordelen en de rol van de componenten leverden direct inzicht en behoefde geen verdere uitwerking.
In het coderingsschema (figuur 8) is een totaaloverzicht te zien van het resultaat van het coderen en verfijnen.



Figuur 8: Coderingsschema

4.3. Resultaten

In totaal zijn 204 coderingen gemaakt en in perspectief geplaatst. Er is steeds bijgehouden waar in het interview de uitspraak is gedaan, in lijn met de interviewstructuur die geëvalueerd is in onderdeel 4.1. De tabel met coderingen is ondergebracht in bijlage 6. Doordat de oorsprong vastgelegd is in de tabel en de betreffende tekst **bold** gemaakt is in bijlage 5, kan relatief eenvoudig de uitspraak in het interview en de bijbehorende context worden teruggevonden.

De vijf geïnterpreteerde determinanten (klant, domein, architectuur, cultuur en testen) worden grotendeels bevestigd in de zes interviews. De belangrijkste nieuwe inzichten zijn dat de keuze voor CDV op een grotendeels onbewuste wijze wordt gemaakt, waar organisaties als vanzelf inrollen zo lang ze aan voldoende vernieuwing doen. De ontwikkeling naar CDV wordt een aantal keer genoemd als IT-push. Daarnaast blijkt dat geen sterke afweging gemaakt wordt over kosten, maar men vooral op zoek is naar de baten en tot een meer kwalitatieve business case komt. De voordelen van CDV spelen een grote rol bij het maken van de adoptiebeslissing en de rol van de componenten komt veel minder sterk naar voren. Het nut van de drie overzichten (voordelen, componenten, determinanten) wordt bevestigd.

5. Discussie, conclusies en aanbevelingen

In dit hoofdstuk wordt dieper ingegaan op de resultaten van het empirisch onderzoek. Er wordt getracht conclusies te trekken uit de verzamelde data, waarbij sterkten en zwakten besproken worden en daarmee de waarde van de conclusies duidelijk moet worden. Tot slot worden aanbevelingen gedaan voor de praktijk en voor verder onderzoek.

5.1. Discussie

Een belangrijk element in het interview is de kernvraag (Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren?). Een open vraag die de deelnemer uitnodigt vrijuit te vertellen. De onderwerpen die de deelnemers naar verwachting zullen aandragen, komen later in het interview aan bod in de vorm van de voordelen, componenten en determinanten. Er zou aan uitspraken die in eerste instantie worden gedaan, meer waarde gehecht kunnen worden dan aan uitspraken die de meer gestuurde vragen bevestigen. Hoe dan ook is het interessant om te kijken of de deelnemers consistent zijn in hun beantwoording, door de antwoorden uit verschillende onderdelen van het interview met elkaar te vergelijken. De tabel in bijlage 6 kan al geanalyseerd worden met eenvoudige spreadsheet software zoals MS Excel, om zo overeenkomsten en verschillen inzichtelijk te maken die niet direct uit hoofdstuk 4 blijken.
Het coderingsschema (figuur 8) laat zien dat de determinanten uit het literatuuronderzoek op veel steun kunnen rekenen in het empirisch onderzoek. Dit verdient wel de nodige nuance, wat hieronder wordt besproken, gevolgd door bespreking van de andere coderingen.

<u>Cultuur</u>
Over cultuur lijkt de minste discussie te bestaan. Maar liefst 22 uitspraken zijn gevonden die de determinant bevestigen, zonder conflicterende of neutrale uitspraak. In alle zes de interviews is onderbouwing te vinden. Wanneer gekeken wordt naar de beantwoording tot en met de kernvraag blijkt dat het zeven van de 22 uitspraken betreft, verdeeld over drie van de zes interviews. De zaken die deelnemers, onder andere belangrijk vonden te bespreken, gaan over de vrijheid die teams hebben bij het kiezen van hun werkwijze, hoe een oude manier van denken plaats (heeft ge)maakt voor een nieuwe mindset, hoe nieuwe medewerkers dit versterken, de autonomie en beslissingsbevoegdheid aanwezig is (coderingen 2, 12, 40, 41, 93, 95).
Consistentie binnen deze interviews (nummer 1, 2 en 4) is dan ook snel gevonden. Bij het bespreken van de determinanten wordt bevestigd wat eerder is aangekaart. Waar in het eerste deel van het interview cultuur niet altijd letterlijk wordt genoemd, wordt bij het bespreken van de determinanten duidelijk dat de deelnemers wel degelijk de cultuur bedoelen. Dit komt terug in de coderingen 29, 54, 55, 57 en 119.
Bij interviews nummer 3 en 6 wordt cultuur naar voren gebracht bij het bespreken van de componenten, wat krachtiger is dan dat het volledig reactief ter sprake komt bij de determinanten. In deze interviews wordt de kritieke waarde van cultuur duidelijk gemaakt. CDV opzetten zonder cultuur van vernieuwing en vertrouwen zou niet mogelijk zijn en wanneer het management geen vertrouwen en zelfstandigheid geeft aan teams, dan zou het kansloos zijn (coderingen 90 en 200). In interview 5 komt cultuur pas bij de determinanten ter sprake. Toch wordt ook daarin niets aan het toeval overgelaten: "*Ik denk cultuur een van de belangrijkste factoren is om rekening mee te houden"* (152).

<u>Domein</u>
Vervolgens lijkt domein de determinant waar ook relatief snel steun voor gevonden kan worden. Hiervoor zijn twaalf uitspraken gedaan en twee neutrale uitspraken. Het moment in de interviews

waarop uitspraken worden gedaan valt wel op. Het domein wordt vooral besproken aan het einde van het interview. Alleen in interviews 1 en 3 wordt eerder in het gesprek naar voren gebracht dat domein een rol speelt en dat zijn geen krachtige uitspraken (10, 67). Men lijkt zich niet erg bewust te zijn van het domein. Wellicht niet vreemd, want het domein kan vooral gezien worden als een classificering, een gegeven dus en niet zozeer iets wat aan te passen is. Deelnemers lijken in eerste instantie niet zo goed te weten wat ze nu moeten met het domein, wat het best terugkomt in codering 195. Daarnaast werd in de interviews al snel de relatie gelegd met architectuur. De beperkingen die een domein met zich meebrengt, wordt eigenlijk in de architectuur gevoeld lijkt het. Dit wordt op treffende wijze besproken in coderingen 149, 85, 34 en 64. Ondanks dat betoogd kan worden dat het domein als determinant voldoende steun vindt, kan het eigenlijk niet los gezien worden van architectuur. Het expliciet uit elkaar trekken van de twee onderwerpen lijkt de bewustwording echter wel ten goede te komen. Bovendien is hetgeen dat besproken en uitgesproken is, in lijn met de opgestelde determinant, waarin vooral wordt gesteld dat het domein bepalend is voor het uitrollen van de software (naar de klant) en dat het grondig geanalyseerd dient te worden. In interview 1 is het domein het meest besproken en wordt goed duidelijk welke invloed het kan hebben.

Architectuur

Bij architectuur komen we voor het eerst conflicterende uitspraken tegen. Puur cijfermatig bekeken is vrij duidelijk dat architectuur overeind blijft als determinant, maar enige verdieping is op zijn plaats. In drie van de zes interviews (nummer 1, 4 en 5) wordt vroegtijdig steun gevonden voor architectuur. Wat duidelijk wordt, is dat de architectuur geschikt moet zijn, of het moet aankunnen. Het is een randvoorwaarde (13, 102, 111). Dit wordt bevestigd bij het bespreken van de determinanten, maar sterke overtuiging ontbreekt hier. Gezien de kritieke rol van architectuur die naar voren kwam in het literatuuronderzoek en is opgenomen in het overzicht van determinanten, was de verwachting dat hierover meer overtuiging te vinden zou zijn in de praktijk. In de coderingen die als neutraal beoordeeld zijn, wordt dit enigszins verduidelijkt (88, 196). Men lijkt zich niet heel bewust te zijn van de kritieke rol en het gaat goed, zo lang het niet in de weg zit.
De conflicterende uitspraken zijn te herleiden tot interview 6. Hierin wordt gesteld dat wanneer een applicatie groot en complex is, dat de flexibiliteit en daarmee CI/CD, in de weg zit. Omvang en complexiteit zouden niet per se gelijk zijn aan architectuur. Daar staat echter tegenover dat hetgeen wat beschreven is over monolithische software in het literatuuronderzoek, erg sterk overeenkomt met hetgeen dat besproken is. Of het nu aan de softwarearchitectuur ligt of aan de complexiteit van de software in het algemeen zou mogelijk verzanden in een definitiekwestie. De uitkomsten van het literatuuronderzoek zijn op dit punt ook zeer uitgebreid en sterk onderbouwd (Chen, 2015b; Humble, 2017; Shahin et al., 2019). De deelnemer gaf bij het bespreken van de determinanten ook wel aan dat architectuur een determinant is, zij het met beperkte overtuiging (197). Wellicht is het meest interessant nog dat sprake is van een implementatie van CDV bij monolithische software, zonder dat daar de architectuur voor omgegooid is (196). Gedurende het interview werd duidelijk dat vooral werd ingezet op CI en niet zozeer op CDV, wat zou kunnen verklaren waarom toch resultaat bereikt is. Uit de context blijkt echter dat het wel degelijk om ver doorgevoerde testautomatisering gaat en geautomatiseerd deployen van software naar alle beschikbare omgevingen. Dit zou dus wel degelijk als CDV gezien kunnen worden, wat bevestigd wordt in codering 179.

Klant

In vier van de zes interviews wordt vroegtijdig gesproken over de rol van de klant. Hiermee lijkt de klant wel degelijk een rol te spelen. In interviews 5 en 6 wordt dit ook duidelijk naar voren gebracht,

zie hiervoor coderingen 140, 163, 169 en 175. Uit deze interviews blijkt ook duidelijk waarom de klant zo belangrijk is om rekening mee te houden. Het is een beperkende factor, omdat bijvoorbeeld sprake is van on-premises software. Klanten moeten hierbij zelf nog een acceptatieproces doorlopen, wat dan nog niet versneld is middels CDV werkwijze (142, 145, 173, 181). Bij de interviews waar de klant geen beperkende factor is, kunnen de deelnemers zich nauwelijks voorstellen dat de klant enige rol speelt. Hier is echt sprake van een volledig ander perspectief. Klanten willen altijd nieuwere software is de overtuiging. Er zijn geen nadelen. Dit komt sterk naar voren in coderingen 83 en 116. Er wordt dan ook door twee van de zes deelnemers gepleit dat de klant geen factor is om rekening mee te houden (58, 116). Een derde deelnemer zit op dezelfde lijn, maar is genuanceerder (83). Hierbij is besproken dat de beperkte rol van de klant te maken zou hebben met het domein. Het betreffen hier namelijk webapplicaties. Deze hebben de eigenschap dat ze zonder enig ongemak van een upgrade voorzien kunnen worden. De klant blijft immers gebruik maken van dezelfde browser. Mogelijk is sprake van een relatie tussen de mindset van professionals en de manier waarop ze de klant bedienen met nieuwe software. Bij interviews 2, 3 en 4 is duidelijk dat niet echt rekening gehouden wordt met de klant en is steeds sprake van het domein webapplicaties. Interviews 1, 5 en 6 brengen een complexer beeld naar voren, waarbij de klant wel degelijk als beperkende factor gezien kan worden omdat de klant zelf ook werk heeft aan een software upgrade. Klant is ook een breed begrip. Een consument die een vakantie boekt op een website heeft andere belangen bij de gebruikte software dan een klant die software gebruikt om de backoffice administratie van hypotheek producten te organiseren. De gedachte dat B2C en B2B diensten hierin verschillen is niet vreemd en het domein webapplicaties bedient over het algemeen B2C diensten.

Testen

Van de vijf determinanten wordt het minst vaak gesproken over testen. Slechts in een interview (nummer 3) wordt vroegtijdig gesproken over de rol van testen (72, 74). Verderop in het interview wordt bevestigd dat testen een randvoorwaarde is (92). De invloed op de adoptiebeslissing komt niet sterk naar voren. Dat is ook in andere interviews terug te lezen. Slechts in een geval wordt gesteld dat testen een belangrijke rol had, maar in de daaropvolgende onderbouwing gaat het eigenlijk ergens anders over (156). Dat is niet heel sterk. Het meest opvallend is dat twee keer gesteld wordt dat de testinspanning niet toeneemt, maar juist afneemt, door automatisering (53, 120). Hierin zit ook het conflicterende element. Alle deelnemers zijn het erover eens dat testen belangrijk is, maar dat de testinspanning toeneemt is niet zeker. Testen is doorgaans al erg belangrijk bij softwareontwikkeling. CDV bestaat voor een belangrijk deel uit testautomatisering, waardoor de inspanning ook minder zou kunnen worden. Testen is dus belangrijk, maar dat er het minst over gesproken is, is mogelijk ook een indicatie voor de zelfsprekendheid daarvan. De rol bij de adoptiebeslissing lijkt eigenlijk beperkt.

Relaties

Bij de interview werden verschillende keren relaties naar voren gebracht. Slechts eenmaal wordt gesproken over hoe de componenten onderling samenhangen. Bij de voordelen en determinanten komen beiden zeven uitspraken naar voren die ervoor pleiten dat de betreffende onderwerpen niet altijd op zichzelf staan. Wat duidelijk lijkt te worden is dat verhoogde klanttevredenheid wordt bereikt, als gevolg van de andere voordelen van CDV. Tussen de determinanten domein en architectuur komt ook relatief vaak een relatie ter sprake, waarbij wel opgemerkt moet worden dat de onderzoeker deze relatie vaak al benoemde, waarop de deelnemers bevestigden. Dat maakt het lastiger hier conclusies aan te verbinden.

Rol voordelen en rol componenten

Over de rol van de voordelen zijn de deelnemers tamelijk eensgezind en duidelijk. De (alle) voordelen worden herkend en speelden een rol bij het maken van de adoptiebeslissing. In ieder interview is een uitspraak gevonden die dit (bijna) letterlijk laat zien (19, 46, 77, 106, 130, 187). Dit leidt niet zozeer tot discussie, maar vooral tot conclusie.

De componenten spreken minder tot de verbeelding. Er was sprake van minder herkenning en in de interviews moesten deelnemers meer op weg geholpen worden. De rol bij de adoptiebeslissing wordt niet zo duidelijk, wat ook kan komen doordat deze componenten meer uitnodigen om de daadwerkelijk implementatie van CDV te bespreken, dan het maken van de beslissing daartoe over te gaan. Deelnemers gaven ook aan onderscheid te zien in de componenten. Daar waar (sterke) herkenning was, gold dat vaak voor enkele componenten en niet voor allemaal. Voor ieder interview komt dit terug, te vinden in coderingen 22, 47, 79, 110, 136 en 192. Dit kan goed schematisch worden weergegeven, waardoor tabel 9 ontstaat.

Component → ↓Interview	1	2	3	4	5	6	7	8	9	10
1	X	X	X	X	X	X	X	X	X	X
2	X		X					X	X	
3	X			X	X					X
4	X	X	X			X				
5	X		X	X	X	X				
6	X					X			X	X
Totaal	**6**	**2**	**4**	**3**	**3**	**4**	**1**	**2**	**3**	**3**

Figuur 9: Scores van de componenten in de interviews

Hieruit blijkt dat snelle en frequente release voor alle deelnemers een duidelijk element van CDV is. Op dit punt is wel herkenning. Voor continuous testing and quality assurance en betrokken klant is ook een meerderheid te vinden die zich aangesproken voelt door de componenten. Voor interview 5 en 6 is dat logisch omdat het in lijn is met wat beschreven is bij de determinant klant. Uit interview 4 blijkt de nuance tussen de klant als determinant en de betrokken klant als component. Deze organisatie was blijkbaar uitstekend is staat om te kiezen voor CDV zonder daarbij echt de klant in overweging te nemen, maar erkent wel dat de klant hier vervolgens in moet worden meegenomen. Wat verder opvalt is dat continue en snelle uitvoering van experimenten eigenlijk niet genoemd wordt. Van de zes interviews, kennen drie de scope van webapplicaties. Een domein waar je deze component juist zou verwachten. Uit de context van de interviews blijkt dat dat ook wel het geval is, maar dat dat niet zozeer gekoppeld is aan de werkwijze van CDV. Organisaties lijken in staat te zijn technieken als A/B testing ook te kunnen omarmen, zonder tot CDV over te gaan.

Doelen

De doelen die naar voren zijn gebracht in de interviews, kennen een sterke samenhang met de voordelen van CDV. De insteek van dit onderzoek was dat een voordeel van CDV organisatie kan helpen met het realiseren van doelen. Doelen en voordelen zijn daarmee niet gelijk aan elkaar. Hoewel deelnemers dit hebben erkend tijdens de interviews, lijkt het maken van dit onderscheid niet veel toe te voegen. De doelen die besproken zijn, kunnen voor het overgrote deel probleemloos gekoppeld worden aan de voordelen die CDV oplevert. Dit is in lijn met de sterke herkenning die deelnemers hebben bij het expliciet bespreken van deze voordelen. Wat dit versterkt, is dat deelnemers, zonder uitzondering, vroeg in het interview naar voren brengen welke doelen de organisatie wil bereiken met de implementatie van CDV. Er is een hoge consistentie te vinden tussen

de drie momenten waarop de doelen worden besproken. De hoofdvraag, het vragen naar de doelen (van de organisatie), het bespreken van de voordelen (van CDV).
Ook zijn drie doelen gevonden die niet zonder meer passen in de bekende voordelen (15, 76, 129). Omdat deze incidenteel genoemd worden, is het lastig er conclusies uit te trekken. Ze zijn het wel waard benoemd te worden. CDV kan worden ingezet om te helpen een aantrekkelijke werkgever te zijn voor nieuw talent, vanwege de flexibele en Agile manier van werken. CDV kan organisaties helpen hun interne beheersing aan te tonen voor bijvoorbeeld de toezichthouder, vanwege de sterke inzet op transparantie en traceerbaarheid. Dit komt ook terug in het literatuuronderzoek (Siqueira et al., 2018). De kennis die in hoofden van medewerkers aanwezig is, waar CDV een einde aan zou maken (129) is een doel in lijn met eerdergenoemde. Deze uitspraak kan probleemloos gekoppeld worden aan de beschreven transparantie en traceerbaarheid.

Nieuwe inzichten

Wellicht het meest interessante onderdeel van dit onderzoek komt terug in wat de deelnemers vertellen over de overwegingen van organisaties om tot CDV over gaan, zonder dat dit direct onder te brengen is in de eerder gevonden voordelen, componenten en determinanten. Wat het meest opvalt, is dat organisaties zich van veel dingen niet bewust zijn ten tijde van de adoptiebeslissing. Later bij de implementatie blijkt dan bijvoorbeeld dat het toch een factor was om rekening mee te houden, of valt het bijna aan het toeval toe te rekenen dat het succesvol is verlopen. Dit blijkt uit de coderingen ‘onbewuste keuze’, waarvan er twaalf gevonden zijn. Uit vijf van de zes interviews komt naar voren dat organisaties hier op min of meer onbewuste manier inrollen, of dat een keuze wordt gemaakt zonder goed te weten wat de consequenties hiervan zijn. Slechts interview vier lijkt hierop een uitzondering te zijn.
Wat hiermee lijkt samen te hangen, is dat de onderzochte organisaties in het algemeen vaak wel de ambitie hebben om zich te ontwikkelen. Hiervoor zijn zeven coderingen gevonden. Een samenhang tussen de algemene ontwikkeling die organisaties maken en de manier waarop ze voor CDV kiezen kunnen met elkaar te maken hebben. De voordelen die CDV biedt, zijn erg interessant voor organisaties. Dit staat los van een expliciete keuze voor het omarmen van de (volledige) werkwijze. Dit komt het beste terug in coderingen 62 en 167. In interview vier wordt duidelijk dat het een organisatie betreft die zorgt dat ze meegaat in de ontwikkelingen van de wereld om hen heen (102-112). Het lijkt erop dat de kans om de voordelen van CDV te benutten, groter wordt naarmate organisaties meegaan met hun tijd en zorgen voor een moderne architectuur en cultuur.
Dat laatste kan nog weleens lastig zijn. Afhankelijk van het domein hebben organisaties hier niet altijd invloed op. Dit komt terug in de code ‘domein bepaalt cultuur’. Hoewel dit bij slechts een organisatie (11, 28, 36) naar voren wordt gebracht, zit er wel bepaalde logica in deze uitspraken. Dat het opnieuw neerzetten van de softwarearchitectuur voor een domein als ERP, lang kan duren en kostbaar is, wordt in het literatuuronderzoek ook al erkend. Wat wel nieuw is, is de mogelijkheid dat niet alleen de architectuur in dergelijk domein verouderd is, maar de manier van denken of de cultuur ook. Deze twee elementen versterken elkaar dan. Voor organisaties die de middelen hebben om dit te doorbreken zou dat een grote kans kunnen zijn. Voor organisaties die maar niet voldoende in beweging komen op dergelijke domeinen zou dat een bedreiging zijn.
Het benutten van een kans versus het tegengaan van een bedreiging komt ook terug in het ‘vooruitlopen of bijblijven’. Dit onderwerp wordt slechts incidenteel naar voren gebracht en was relevant in het betreffende interview, omdat de keuze voor CDV daar al relatief lang geleden was. Dat zou meer als het benutten van een kans gezien kunnen worden, gepaard met relatief veel risico en onzekerheid. Inmiddels zou meer bekend moeten zijn over de risico’s en onzekerheden van CDV.
De factor kosten komt in drie interviews ter sprake. Op zichzelf is dat al opvallend, omdat CDV in potentie hoge kosten met zich meebrengt. Organisaties lijken echter niet in staat om (vooraf) een

overzicht te maken van de kosten en baten van de investering. De baten (voordelen) staan echter wel scherp op het netvlies. Het lijkt erop dat binnen organisaties mensen overtuigd kunnen worden van de voordelen van CDV, zonder onderbouwing van een (kwantitatieve) business case. Op meer kwalitatieve wijze komt men dan tot de overtuiging dat de introductie van CDV meer waarde uit bestaande resources en inspanningen zal halen.
Dit komt ook enigszins terug in de coderingen voor 'overtuigen' (50, 51), al betreft dat slechts een interview. Verder is daar niet iets uit te concluderen.
Waar mogelijk wel een conclusie uit getrokken kan worden, is het feit dat bij drie interviews expliciet wordt aangegeven dat de ontwikkeling van CDV vanuit IT komt (IT-push, 69, 78, 94, 122, 154). In interview 6 wordt dit niet expliciet naar voren gebracht, maar dit betreft een IT organisatie die software levert aan andere organisaties. Hierbij wordt opgemerkt dat de klant nog weleens de beperkende factor is. De uitspraken die worden gedaan suggereren dat de kans om de voordelen van CDV te benutten, vaak aan de IT kant van organisaties wordt ontdekt. Wat daar mogelijk mee samenhangt is dat CDV wordt gezien als een manier om (meer) Agile te werken. Dit zou kunnen betekenen dat IT onder druk van een steeds veeleisender omgeving, op zoek gaat naar manier om sneller meer waarde op te leveren en zodoende uitkomt op CDV, eerder dan dat de business bij IT aanklopt om concreet om de werkwijze van CDV te vragen.
Deze relatie tussen business en IT zien we complexer worden op het momenten dat deze twee verdeeld zijn over verschillende organisaties. In interviews 1 en 6 zien we dat terugkomen, daar is in beide gevallen sprake van een relatie tussen leverancier en klant. Uit de context van interview 5 blijkt ook dat de relatie tussen klant en leverancier de situatie complexer kan maken. Interview 1 laat zien dat de invloed van de klant op verschillende leveranciers beperkt kan zijn. Daar waar het domein en de architectuur het toelaten, wordt CDV reeds toegepast. Daar waar dat niet zo is, kan de organisatie dat zelf eigenlijk niet veranderen. In interview 6 zien we dat het andersom ook complexer kan worden. De organisatie in kwestie is een softwareleverancier en is relatief slecht in staat om gebruik te maken van productiewaardige omgevingen. Dit bemoeilijkt de implementatie van CDV. In de bestaande literatuur wordt dat al benoemd. Laukkanen et al. (2017) stellen dat verspreide teams een probleem vormen bij de implementatie van CDV. Dit is vrij generiek. Teams kunnen op allerlei manier verspreid zijn en het is denkbaar dat niet alle organisaties hier op dezelfde manier last van hebben. De interviews geven mogelijk genoeg reden specifieker in te gaan op de relatie tussen klant en leverancier en de beperkte invloed die partijen over en weer hebben.
De organisatorische factoren die genoemd worden, zijn van allerlei aard. Het gaat steeds om incidentele opmerkingen die gecombineerd geen stand houden. Hier komen geen nieuwe bevindingen uit naar voren.
Hetzelfde geldt voor de opmerking over greenfield toepassingen versus bestaande toepassingen. Dit onderwerp wordt al benoemd in het literatuuronderzoek. Het was te verwachten dat het in de empirie bevestigd zou worden. Dit gebeurt in slechts een geval en dat is weinig. Daarnaast geldt dat het in het literatuuronderzoek sloeg op de architectuur en in het interview werd benadrukt dat het verschil zit in de inspanning voor testautomatisering.

<u>Nut overzicht</u>
Voor de drie overzichten die opgesteld zijn in het literatuuronderzoek en empirisch getoetst zijn (voordelen, componenten en determinanten), is het interessant om te weten hoe ze gebruikt kunnen worden. Hiervoor zijn vijftien uitspraken gevonden, doorgaans gegeven na expliciete vragen. Deelnemers zijn eigenlijk unaniem over het nut van de overzichten. Ze kunnen goed helpen bij het voeren van de discussie binnen organisaties over de adoptiebeslissing tot CDV over te gaan. In interview 4 komt terug dat de organisatie de benodigde kennis eigenlijk wel had en dat deze overzichten geen invloed zouden hebben gehad op de besluitvorming. Het nut van de overzichten

wordt echter wel erkend. In andere interviews zien we veel sterkere bevestiging van het nut terugkomen. Coderingen 52, 82, 143 en 204 laten dat goed zien. In interview 5 wordt zelfs gesteld dat deze kennis voor een andere situatie had kunnen zorgen. Daarbij moet echter wel meegewogen worden dat deze organisatie relatief lang geleden begonnen is met de implementatie van CDV. De organisatie in interview 4 heeft nog maar kort geleden de beslissing gemaakt conform CDV te gaan werken.

5.2. Conclusies

Bij het maken van de adoptiebeslissing, al dan niet tot CDV over te gaan, komen verschillende onderwerpen kijken. Het literatuuronderzoek leverde componenten, voordelen en determinanten op. In de praktijk blijkt dat bij het maken van de adoptiebeslissing, de onderzochte organisaties zich erg bewust zijn van de voordelen die CDV biedt. De organisaties hebben een helder beeld van de doelen die ze met CDV willen bereiken. Er bestaat echter minder bewustzijn over de componenten van CDV. Hierdoor lijken organisaties goed te snappen waarom ze CDV willen, maar mogelijk niet altijd wat ze nu moeten doen om het een succes te maken.
Voor de in het literatuuronderzoek geïnterpreteerde determinanten wordt in de onderzochte organisaties bevestiging gevonden. Cultuur is daarbij duidelijk de belangrijkste. Van domein en architectuur is men zich minder bewust, maar zo lang men hier niet tegen problemen aanloopt, valt dat niet op. Bij de factor klant zijn interessante verschillen gevonden, die er vooralsnog op wijzen dat de stap naar CDV binnen het domein webapplicaties gemaakt wordt zonder rekening te houden met de klant, terwijl dat voor andere domeinen niet kan. Testen is als factor het minst besproken. Hoewel het belang ervan wel bevestigd wordt door de deelnemers, lijkt het niet echt een rol te spelen bij het maken van de beslissing. Er zijn nieuwe inzichten opgedaan, maar daarbij zijn geen nieuwe determinanten bepaald.
De beslissing om tot CDV over te gaan, lijkt voor de onderzochte organisaties op een onbewuste manier tot stand te komen. Daar waar de keuze voor de werkwijze bewust wordt gemaakt, is men zich doorgaans niet goed bewust van randvoorwaardelijke zaken, ook al zijn ze op orde. CDV lijkt eerder iets waar organisaties als vanzelf inrollen, zo lang ze gewoon meedoen met ontwikkeling en vernieuwing. Hierbij komt een IT-push relatief vaak voor in de onderzochte organisaties.

5.3. Aanbevelingen voor de praktijk

De drie overzichten van componenten, voordelen en determinanten kunnen een sterk middel zijn voor organisaties om hun overwegingen om eventueel tot CDV over te gaan, te structureren en samen te vatten. Van een belangrijk deel van deze overweging lijken organisaties zich nog onvoldoende bewust te zijn. Hoewel toekomstige verbetering van deze overzichten niet ondenkbaar is, staat niets in de weg ze te gebruiken. Organisaties zouden per direct bewust gemaakt moeten worden van het bestaan van deze overzichten. De voordelen beschrijven daarbij waarom organisaties CDV zouden willen. De componenten beschrijven wat CDV is en de determinanten benadrukken waar extra goed op gelet moet worden.
Organisaties dienen daarbij te beseffen dat het niet vreemd is om geen sterke (kwantitatieve) business case uit te werken. Wel dient een grondige analyse gemaakt te worden van de samenhang tussen klant, domein en architectuur. Daarbij kan een inschatting gemaakt worden of sprake is van het benutten van een kans, of het voorkomen van een bedreiging. Tevens moet in de analyse worden meegenomen hoe de relaties liggen tussen klanten en/of leveranciers, wat de omvang is van eventueel betrokken andere organisaties en welke invloed en slagkracht de betreffende eigen organisatie daarbij heeft. Een relatief kleine organisatie die software afneemt van een leverancier,

binnen een verouderd applicatiedomein, zal hoogstwaarschijnlijk niet de invloed en de middelen hebben om CDV te introduceren.

5.4. Reflectie

Dit onderzoek kent sterke punten en beperkingen. Allereerst is overduidelijk dat uit empirisch onderzoek, waarbij zes deelnemers zijn geïnterviewd, geen harde conclusies getrokken kunnen worden. Dit komt reeds terug in de formulering van de discussie en conclusies, maar mag nog expliciet vermeld worden. Voor zover bekend is dit de eerste keer dat met focus op de adoptiebeslissing voor CDV, dergelijke determinanten zijn opgesteld en getoetst. Hoewel de uitkomst bevestigend is, dient het vooral gezien te worden als steun in de rug om in de praktijk gebruikt te gaan worden, maar nog zeker niet als absolute waarheid. De determinanten zijn namelijk door de onderzoeker naar voren gebracht in het interview, waarna bevestigend werd gereageerd. Dit zet de validiteit van het onderzoek enigszins onder druk. Het zou interessant zijn om het onderzoek opnieuw te doen, met vier of zes determinanten in plaats van vijf, of met geheel andere determinanten. Daarmee zou onderzocht worden hoe kritisch deelnemers hierin kunnen zijn en in welke mate ze zich laten leiden door de onderzoeker. Sterk punt dat hiervoor compenseert is dat de interviews startten met een open vraag, waarop de deelnemers de kans kregen min of meer spontaan determinanten naar voren te brengen. In de discussie komt dat ook terug. Dezelfde onderwerpen komen op verschillende momenten en op verschillende manier aan bod.
Interview zes valt op omdat de deelnemer het primair over CI heeft en niet zozeer over CDV. Uit de context blijkt dat de manier van werken ook als CDV uitgelegd kan worden. Toch roept dit de vraag op wat er nu eigenlijk gemeten wordt. Mogelijk zijn de voordelen, componenten en determinanten van CDV in bepaalde mate uitwisselbaar met CI, of met ASD in het algemeen. In het literatuuronderzoek is al naar voren gekomen dat CI, CDV en CDP in publicaties niet altijd hetzelfde betekenen. Om te voorkomen dat dit een probleem werd bij dit onderzoek, is dit aan het begin van ieder interview toegelicht. Dit heeft niet tot discussie of onbegrip geleid en versterkt daarom zowel de validiteit als de betrouwbaarheid van de resultaten. Aanvankelijk was het plan om twee deelnemers per organisatie te vinden, ter bevordering van de betrouwbaarheid. Dit idee is losgelaten. Enerzijds zet dit de betrouwbaarheid onder druk, anderzijds zijn nu meer organisaties betrokken wat een positief effect heeft op de betrouwbaarheid. Tussen het afnemen van het interview en het uitwerken ervan zit relatief lange tijd. Hierdoor zijn de deelnemers niet (meer) benaderd om hun uitspraken te bevestigen. Van de interviews zijn opnamen gemaakt met goede audiokwaliteit en redelijke beeldkwaliteit. Deze opnamen zijn grondig geanalyseerd en de kans op interpretatiefouten wordt minimaal geacht. Het gebrek aan bevestiging achteraf is wel een afwijking op het plan en blijft in zekere mate hoe dan ook een zwak punt in het onderzoek.
De deelnemers zijn conform plan gewezen op de ethische aspecten van het onderzoek en de onderzoeker is overtuigd van een verantwoorde manier van handelen.

5.5. Aanbevelingen voor verder onderzoek

Uit onderdeel 5.1 blijken verschillende interessante zaken die naar voren zijn gekomen tijdens de interviews. Uit een deel hiervan kunnen niet direct conclusies worden getrokken.
Zo zijn relaties ter sprake gekomen, met name tussen de voordelen en tussen de determinanten. Het in kaart brengen van deze relaties was niet de kern van dit onderzoek. Het kan echter wel van toegevoegde waarde zijn deze relaties te doorgronden. Met name tussen de voordelen onderling, de componenten onderling en tussen de voordelen en componenten. Met andere woorden: wat moet een organisatie doen om welk voordeel te bereiken? Hiervoor lijkt diepgaand onderzoek in de vorm van case studies het meest geschikt.

Een mogelijk achtste voordeel kan gevonden worden in de aantoonbaarheid van de interne beheersing, middels transparantie en traceerbaarheid. Organisaties die onder toezicht staan, bijvoorbeeld in de financiële sector, kunnen hier veel baat bij hebben. Grondiger onderzoek bij dergelijke organisaties zou ertoe kunnen leiden dat de voordelen van CDV worden uitgebreid op dit punt. Dit zou ook middels kwantitatief onderzoek getoetst kunnen worden, waarbij op veel grotere schaal onderscheid tussen sectoren en onderscheid in wel of geen behoefte aan transparantie en traceerbaarheid, inzichtelijk gemaakt kan worden.
De centrale rol die het domein lijkt te spelen, dient verder onderbouwd te worden. Wat het applicatiedomein nu precies is en welke domeinen er zijn, is eigenlijk niet helder. Onderzoek hiernaar lijkt terecht, teneinde vast te stellen in welke mate het domein bepalend is voor de determinanten klant, architectuur en cultuur. Hier zou grondig en uitgebreid onderzoek naar gedaan kunnen worden. Zien we bijvoorbeeld een verouderde architectuur en cultuur bij oude domeinen? Zijn er ook oude domeinen die wel aan vernieuwing hebben gedaan en waar dit niet voor geldt? Kwalitatief onderzoek in de vorm van case studies lijkt hier voorlopig het juiste middel voor te zijn, gezien de onzekerheid waarmee dit punt naar voren komt in dit onderzoek.
De relatie tussen klant en leverancier en de positie die de betreffende organisatie hierbij heeft, lijkt een belangrijke invloed te hebben op de implementatie van CDV. Verder onderzoek is nodig om dit vast te stellen en de determinanten hier mogelijk op uit te breiden. Verschillende case studies waarbij verschillende relaties tussen klant en leverancier duidelijk worden, zou hiervoor een geschikt middel kunnen zijn. Eveneens kan daarbij het verschil tussen B2B en B2C diensten naar voren komen. Dit verschil is mogelijk zeer bepalend voor hoe er naar de factor klant gekeken wordt. De klant lijkt niet echt een factor om rekening mee te houden wanneer sprake is van webapplicaties. Dit zou gezien kunnen worden als een relatie tussen klant en domein. Welke andere relaties kunnen gevonden worden tussen klant en domein?
In onderdeel 5.4 komen zwakke punten van dit onderzoek naar voren. Verder onderzoek zou deze punten kunnen wegnemen. Het meest eenvoudige vervolg zou zijn om dit onderzoek uit te breiden met meer interviews. Middels kwantitatief onderzoek zouden veel grotere aantallen organisaties en professionals onderzocht kunnen worden. Daarbij zou een questionnaire ontwikkeld kunnen worden waarmee getracht wordt de zwakheden in 5.4 zoveel mogelijk te verhelpen.

Referenties

Akerele, O. (2018). System dynamics modelling of the impact of agile practice on the quality of continuous delivery projects. *Innovations in Systems and Software Engineering, 14*(3), 183-208. doi:10.1007/s11334-017-0296-z

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). Manifesto for Agile Software Development.

Chen, L. (2015a). Continuous Delivery: Huge Benefits, but Challenges Too. *IEEE Software, 32*(2), 50-54. doi:10.1109/MS.2015.27

Chen, L. (2015b, 2015). *Towards Architecting for Continuous Delivery*.

Chen, L. (2017). Continuous Delivery: Overcoming adoption challenges. *The Journal of Systems & Software, 128*, 72-86. doi:10.1016/j.jss.2017.02.013

Claps, G. G., Berntsson Svensson, R., Aurum, A., Institutionen för data- och, i., fakulteten, I. T., Göteborgs, u., . . . Faculty, I. T. (2015). On the journey to continuous deployment: Technical and social challenges along the way. *Information and Software Technology, 57*, 21-31. doi:10.1016/j.infsof.2014.07.009

Cooper, R. B., & Zmud, R. W. (1990). Information Technology Implementation Research: A Technological Diffusion Approach. *Management Science, 36*(2), 123-139. doi:10.1287/mnsc.36.2.123

Dingsøyr, T., & Lassenius, C. (2016). Emerging themes in agile software development: Introduction to the special section on continuous value delivery. *Information and Software Technology, 77*, 56-60. doi:10.1016/j.infsof.2016.04.018

Eck, A., Uebernickel, F., & Brenner, W. (2014). *Fit for Continuous Integration: How Organizations Assimilate an Agile Practice*.

Humble, J. (2010, 13 augustus). *Continuous Delivery vs Continuous Deployment*. Retrieved from https://continuousdelivery.com/2010/08/continuous-delivery-vs-continuous-deployment/

Humble, J. (2017). Continuous Delivery Sounds Great, but Will It Work Here? *Queue, 15*(6), 57-76. doi:10.1145/3178368.3190610

Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*: Addison-Wesley Professional.

Laukkanen, E., Itkonen, J., & Lassenius, C. (2017). Problems, causes and solutions when adopting continuous delivery—A systematic literature review. *Information and Software Technology, 82*, 55-79. doi:https://doi.org/10.1016/j.infsof.2016.10.001

Leppanen, M., Makinen, S., Pagels, M., Eloranta, V.-P., Itkonen, J., Mantyla, M. V., & Mannisto, T. (2015). The highways and country roads to continuous deployment. *IEEE Software, 32*(2), 64-72. doi:10.1109/MS.2015.50

Neely, S., & Stolt, S. (2013, 5-9 Aug. 2013). *Continuous Delivery? Easy! Just Change Everything (Well, Maybe It Is Not That Easy).* Paper presented at the 2013 Agile Conference.

Rodríguez, P., Haghighatkhah, A., Lwakatare, L. E., Teppola, S., Suomalainen, T., Eskeli, J., . . . Oivo, M. (2017). Continuous deployment of software intensive products and services: A systematic mapping study. *Journal of Systems and Software, 123*, 263-291. doi:https://doi.org/10.1016/j.jss.2015.12.015

Rogers, E. M. (2003). *Diffusion of Innovations* (Fifth, Free Press trade paperback ed.). Riverside: Free Press.

Saunders, M., Lewis, P., & Thornhill, A. (2016). *Research methods for business students: 7. ed*. Harlow u.a: Pearson.

Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. *IEEE Access, 5*, 3909-3943. doi:10.1109/ACCESS.2017.2685629

Shahin, M., Zahedi, M., Babar, M. A., & Zhu, L. M. (2019). An empirical study of architecting for continuous delivery and deployment. *EMPIRICAL SOFTWARE ENGINEERING, 24*(3), 1061-1108. doi:10.1007/s10664-018-9651-4

Siqueira, R., Camarinha, D., Wen, M., Meirelles, P., & Kon, F. (2018). Continuous Delivery: Building Trust in a Large-Scale, Complex Government Organization. *IEEE Software, 35*(2), 38-43. doi:10.1109/MS.2018.111095426

Ståhl, D., & Bosch, J. (2013). *Experienced benefits of continuous integration in industry software product development: A case study*.

Tornatzky, L. G., & Fleischer, M. (1990). *The processes of technological innovation*. United States.

Verschuren, P., & Doorewaard, H. (2007). *Het ontwerpen van een onderzoek* (4 ed.).

Bijlage 1: Eerste zoekresultaat met gebruikte en uitgesloten literatuur

Auteur(s) en jaar	Titel	Beoordeling	Resultaat
Stahl, D., Hallen, K., & Bosch, J. (2017)	Achieving traceability in large scale continuous integration and delivery deployment, usage and validation of the eiffel framework.	Traceerbaarheid in softwareontwikkeling staat hierin centraal.	Niet meenemen in onderzoek
Colomo-Palacios, R., Fernandes, E., Soto-Acosta, P., & Larrucea, X. (2018)	A case analysis of enabling continuous software deployment through knowledge management.	Kennismanagement wordt neergezet als belangrijke enabler bij de implementatie van continuous practices.	Niet meenemen in onderzoek
Ståhl, D., & Bosch, J. (2017)	Cinders: The continuous integration and delivery architecture framework.	Introduceert een architectuur raamwerk dat gebruikt kan worden bij de implementatie van CI/CDV.	Wel meenemen in onderzoek
Humble, J. (2017)	Continuous Delivery Sounds Great, but Will It Work Here?	Gaat in op de bezwaren tegen CDV en hoe daarmee om te gaan.	Wel meenemen in onderzoek
Siqueira, R., Camarinha, D., Wen, M., Meirelles, P., & Kon, F. (2018)	Continuous Delivery: Building Trust in a Large-Scale, Complex Government Organization.	Gaat over uitdagingen en voordelen van CDV.	Wel meenemen in onderzoek
Wettinger, J., Breitenbücher, U., Falkenthal, M., & Leymann, F. (2017)	Collaborative gathering and continuous delivery of DevOps solutions through repositories.	Gaat over de technische kant van implementatie, specifiek over repositories.	Niet meenemen in onderzoek
Akerele, O. (2018)	System dynamics modelling of the impact of agile practice on the quality of continuous delivery projects.	Gaat in op de impact van bepaalde oplossingen op de kwaliteit van de CDV implementatie. Daarbij is een model ontwikkeld	Wel meenemen in onderzoek
Chen, L. (2015)	Continuous Delivery: Huge Benefits, but Challenges Too.	Gaat in op de voordelen van CDV	Wel* meenemen in onderzoek
Chen, L. (2017)	Continuous Delivery: Overcoming adoption challenges.	Gaat in op het overwinnen van de uitdagingen van CDV implementatie	Wel meenemen in onderzoek
Claps, G. G., Berntsson Svensson, R., Aurum, A., Institutionen för data- och,	On the journey to continuous deployment:	Gaat in op technische en sociale	Wel* meenemen in onderzoek

i., fakulteten, I. T., Göteborgs, u., . . . Faculty, I. T. (2015)	Technical and social challenges along the way.	uitdagingen van CDP implementatie	
Dingsøyr, T., & Lassenius, C. (2016)	Emerging themes in agile software development: Introduction to the special section on continuous value delivery.	Gaat over de relatie tussen klant en leverancier bij ASD. Zou interessante input kunnen zijn voor de gezochte factoren.	Wel meenemen in onderzoek
Laukkanen, E., Itkonen, J., & Lassenius, C. (2017)	Problems, causes and solutions when adopting continuous delivery—A systematic literature review.	Uitgebreide SLR, overzicht van relatief veel publicaties.	Wel meenemen in onderzoek
Leppanen, M., Makinen, S., Pagels, M., Eloranta, V.-P., Itkonen, J., Mantyla, M. V., & Mannisto, T. (2015)	The highways and country roads to continuous deployment.	Gaat over 15 bedrijven en hun adoptie van CDP.	Wel* meenemen in onderzoek
Mårtensson, T., Ståhl, D., & Bosch, J. (2017)	Exploratory testing of large-scale systems – Testing in the continuous integration and delivery pipeline.	Gaat specifiek over exploratief testen.	Niet meenemen in onderzoek
Mårtensson, T., Ståhl, D., & Bosch, J. (2019)	Test activities in the continuous integration and delivery pipeline.	Gaat specifiek in op testen.	Niet meenemen in onderzoek
Mascheroni, M. A., & Irrazábal, E. (2018)	Continuous Testing and Solutions for Testing Problems in Continuous Delivery: A Systematic Literature Review	SLR waarbij testen centraal wordt gesteld	Niet meenemen in onderzoek
Niu, N., Brinkkemper, S., Franch, X., Partanen, J., & Savolainen, J. (2018)	Requirements Engineering and Continuous Deployment.	Gaat in op requirements engineering in het CDV tijdperk	Niet meenemen in onderzoek
Parnin, C., Helms, E., Atlee, C., Boughton, H., Ghattas, M., Glover, A., . . . Williams, L. (2017)	The Top 10 Adages in Continuous Deployment.	Tien uitspraken over CDP die een werkende set aan instructie moeten vormen, empirisch te toetsen	Wel meenemen in onderzoek
Prieto, I., Izkara, J. L., & Béjar, R. (2018)	A continuous deployment-based approach for the collaborative creation, maintenance, testing and deployment of CityGML models.	Gaat specifiek over 3D modellen van steden	Niet meenemen in onderzoek

Rodríguez, P., Haghighatkhah, A., Lwakatare, L. E., Teppola, S., Suomalainen, T., Eskeli, J., . . . Oivo, M. (2017)	Continuous deployment of software intensive products and services: A systematic mapping study.	Uitgebreide SLR, overzicht van relatief veel publicaties.	Wel meenemen in onderzoek
Schleicher, J. M., Vögler, M., Inzinger, C., & Dustdar, S. (2016)	Smart Brix—a continuous evolution framework for container application deployments.	Gaat specifiek over deployments met behulp van containers.	Niet meenemen in onderzoek
Shahin, M., Babar, M. A., & Zhu, L. (2017)	Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices.	Uitgebreide SLR, overzicht van relatief veel publicaties.	Wel meenemen in onderzoek
Shahin, M., Zahedi, M., Babar, M. A., & Zhu, L. M. (2019)	An empirical study of architecting for continuous delivery and deployment.	Gaat in op de software architectuur die vereist zou zijn voor implementatie van CDV/CDP	Wel meenemen in onderzoek
Shahzeydi, M., Javdani, T., & Sadeghi, R. (2018)	Quality Aspects of Continuous Delivery in Practice.	Gaat specifiek in op kwaliteitsfactoren van CDV. Magere publicatie	Niet meenemen in onderzoek
Srinivasan, S. M. C. S. S. R. D. V. R. P. K. (2013)	Continuous maintenance quality improvement using analytic maintenance quality function deployment technique.	Gaat over onderhoud in de industrie, niet zozeer over software	Niet meenemen in onderzoek

Literatuur gevonden middels sneeuwbalmethode:

Abrahamsson, P., Ebert, C., & Oza, N. (2012). Lean Software Development. *IEEE Software, 29*, 22-25. doi:10.1109/MS.2012.116

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). Manifesto for Agile Software Development.

Chen, L. (2015, 2015). *Towards Architecting for Continuous Delivery*.

Eck, A., Uebernickel, F., & Brenner, W. (2014). *Fit for Continuous Integration: How Organizations Assimilate an Agile Practice*.

Humble, J. (2010, 13 augustus). *Continuous Delivery vs Continuous Deployment*. Retrieved from https://continuousdelivery.com/2010/08/continuous-delivery-vs-continuous-deployment/

Humble, J., & Farley, D. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*: Addison-Wesley Professional.

Neely, S., & Stolt, S. (2013, 5-9 Aug. 2013). *Continuous Delivery? Easy! Just Change Everything (Well, Maybe It Is Not That Easy).* Paper presented at the 2013 Agile Conference.

Ståhl, D., & Bosch, J. (2013). *Experienced benefits of continuous integration in industry software product development: A case study*.

Bijlage 2: Aanvullend zoekresultaat met uitgesloten literatuur

Publicatie
Adams, B., Bellomo, S., Bird, C., Debic, B., Khomh, F., Moir, K., & Oduinn, J. (2018). Release Engineering 3.0. *IEEE Software, 35*(2), 22-25. doi:10.1109/MS.2018.1661327
Adams, B., Bellomo, S., Bird, C., Marshall-Keim, T., Khomh, F., & Moir, K. (2015). The Practice and Future of Release Engineering: A Roundtable with Three Release Engineers. *IEEE Software, 32*(2), 42-49. doi:10.1109/MS.2015.52
Arcilla, R., Brown, D., & Herman, M. (2014). Continuous Integration and Deployment Software to Automate Nuclear Data Verification and Validation. *Nuclear Data Sheets, 118*, 422-425. doi:10.1016/j.nds.2014.04.096
Bass, L. (2018). The Software Architect and DevOps. *IEEE Software, 35*(1), 8-10. doi:10.1109/MS.2017.4541051
Behere, S., Törngren, M., Inbyggda, s., Maskinkonstruktion, Skolan för industriell teknik och, m., & Kth. (2016). A functional reference architecture for autonomous driving. *Information and Software Technology, 73*, 136-150. doi:10.1016/j.infsof.2015.12.008
Bosch, J. (2010). Architecture in the Age of Compositionality. In (Vol. 6285, pp. 1-4). Berlin, Heidelberg: Springer Berlin Heidelberg.
Bruegge, B., Krusche, S., & Alperowitz, L. (2015). Software Engineering Project Courses with Industrial Clients. *ACM Transactions on Computing Education (TOCE), 15*(4), 1-31. doi:10.1145/2732155
Brunnert, A., & Krcmar, H. (2017). Continuous performance evaluation and capacity planning using resource profiles for enterprise applications. *The Journal of Systems & Software, 123*, 239-262. doi:10.1016/j.jss.2015.08.030
Buryak, Y. I. (2010). Continuous performance monitoring of industrial products as a key to their safe delivery and operation. *Automation and Remote Control, 71*(10), 2186-2194. doi:10.1134/S0005117910100206
Callanan, M., & Spillane, A. (2016). DevOps: Making It Easy to Do the Right Thing. *IEEE Software, 33*(3), 53-59. doi:10.1109/MS.2016.66
Cantor, M., & Royce, W. (2014). Economic Governance of Software Delivery. *IEEE Software, 31*(1), 54-61. doi:10.1109/MS.2013.102
Carver, J. C., & Serebrenik, A. (2017). ICSE Highlights. *IEEE Software, 34*(6), 18-20. doi:10.1109/MS.2017.4121213
Cha, B., Kim, J., Moon, H., & Pan, S. (2017). Global experimental verification of Docker-based secured mVoIP to protect against eavesdropping and DoS attacks. *EURASIP Journal on Wireless Communications and Networking, 2017*(1), 1-14. doi:10.1186/s13638-017-0843-1
Chen, J., Xu, X., Osterweil, L. J., Zhu, L., Brun, Y., Bass, L., . . . Wang, Q. (2015). Using simulation to evaluate error detection strategies: A case study of cloud-based deployment processes. *The Journal of Systems & Software, 110*, 205-221. doi:10.1016/j.jss.2015.08.043
Chitic, S.-G., Couturier, B., Clemencic, M., & Closier, J. (2019). LHCb Continuous Integration and Deployment system: a message based approach. *EPJ Web of Conferences, 214*, 5001. doi:10.1051/epjconf/201921405001
Cruz, L., Abreu, R., & Lo, D. (2019). To the attention of mobile software developers: guess what, test your app. *EMPIRICAL SOFTWARE ENGINEERING, 24*(4), 2438-2468. doi:10.1007/s10664-019-09701-0
Denning, S. (2015). Haydn Shaughnessy: understanding the shift to the new economy. *Strategy & Leadership, 43*(3), 37-43. doi:10.1108/SL-03-2015-0022
Denning, S. (2015). New lessons for leaders about continuous innovation. *Strategy & Leadership, 43*(1), 11-15. doi:10.1108/SL-11-2014-0083
El Beggar, O. (2018). Multicriteria decision aid for agile methods evaluation using fuzzy PROMETHEE. *Journal of Software: Evolution and Process, 30*(12), e2108-n/a. doi:10.1002/smr.2108
Gong, Y., & Janssen, M. (2015). Demystifying the benefits and risks of Lean service innovation: a banking case study. *Journal of Systems and Information Technology, 17*(4), 364-380. doi:10.1108/JSIT-03-2015-0019
Gonzalez-Herrera, I., Bourcier, J., Daubert, E., Rudametkin, W., Barais, O., Fouquet, F., . . . Baudry, B. (2016). ScapeGoat: Spotting abnormal resource usage in component-based reconfigurable software systems. *The Journal of Systems & Software, 122*, 398-415. doi:10.1016/j.jss.2016.02.027

Gupta, V., Kapur, P. K., & Kumar, D. (2017). Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling. *Information and Software Technology, 92*, 75-91. doi:10.1016/j.infsof.2017.07.010

Hemon, A., Lyonnet, B., Rowe, F., & Fitzgerald, B. (2019). From Agile to DevOps: Smart Skills and Collaborations. *Information Systems Frontiers*. doi:10.1007/s10796-019-09905-1

Jabangwe, R., Edison, H., & Duc, A. N. (2018). Software engineering process models for mobile app development: A systematic literature review. *The Journal of Systems & Software, 145*, 98-111. doi:10.1016/j.jss.2018.08.028

Karvonen, T., Behutiye, W., Oivo, M., & Kuvaja, P. (2017). Systematic literature review on the impacts of agile release engineering practices. *Information and Software Technology, 86*, 87-100. doi:10.1016/j.infsof.2017.01.009

Knauss, E., fakulteten, I. T., Göteborgs, u., Gothenburg, U., Faculty, I. T., & Institutionen för data- och informationsteknik, S. E. (2019). The Missing Requirements Perspective in Large-Scale Agile System Development. *IEEE Software, 36*(3), 9-13. doi:10.1109/MS.2019.2896875

Krancher, O., Luther, P., & Jost, M. (2018). Key Affordances of Platform-as-a-Service: Self-Organization and Continuous Feedback. *Journal of Management Information Systems, 35*(3), 776-812. doi:10.1080/07421222.2018.1481636

Laukkanen, E., Paasivaara, M., Itkonen, J., & Lassenius, C. (2018). Comparison of release engineering practices in a large mature company and a startup. *EMPIRICAL SOFTWARE ENGINEERING, 23*(6), 3535-3577. doi:10.1007/s10664-018-9616-7

Leppanen, M., Makinen, S., Lahtinen, S., Sievi-Korte, O., Tuovinen, A.-P., & Mannisto, T. (2015). Refactoring-a Shot in the Dark? *IEEE Software, 32*(6), 62-70. doi:10.1109/MS.2015.132

Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., . . . Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology, 114*, 217-230. doi:10.1016/j.infsof.2019.06.010

Mäkinen, S., Leppänen, M., Kilamo, T., Mattila, A.-L., Laukkanen, E., Pagels, M., & Männistö, T. (2016). Improving the delivery cycle: A multiple-case study of the toolchains in Finnish software intensive enterprises. *Information and Software Technology, 80*, 175-194. doi:10.1016/j.infsof.2016.09.001

Mansfield-Devine, S. (2018). DevOps: finding room for security. *Network Security, 2018*(7), 15-20. doi:10.1016/S1353-4858(18)30070-9

Mårtensson, T., Ståhl, D., & Bosch, J. (2018). Enable more frequent integration of software in industry projects. *The Journal of Systems & Software, 142*, 223-236. doi:10.1016/j.jss.2018.05.002

Martin, A., Raponi, S., Combe, T., & Di Pietro, R. (2018). Docker ecosystem – Vulnerability Analysis. *Computer Communications, 122*, 30-43. doi:10.1016/j.comcom.2018.03.011

Meyer, M. (2014). Continuous Integration and Its Tools. *IEEE Software, 31*(3), 14-16. doi:10.1109/MS.2014.58

Ozkaya, I. (2019). Are DevOps and Automation Our Next Silver Bullet? *IEEE Software, 36*(4), 3-95. doi:10.1109/MS.2019.2910943

Perez-Palacin, D., Merseguer, J., Requeno, J. I., Guerriero, M., Di Nitto, E., & Tamburri, D. A. (2019). A UML Profile for the Design, Quality Assessment and Deployment of Data-intensive Applications. *Software and Systems Modeling, 18*(6), 3577-3614. doi:10.1007/s10270-019-00730-3

Poppendieck, M., & Cusumano, M. A. (2012). Lean Software Development: A Tutorial. *IEEE Software, 29*(5), 26-32. doi:10.1109/MS.2012.107

Prikladnicki, R., Lassenius, C., & Carver, J. C. (2018). Trends in Agile Updated: Perspectives from the Practitioners. *IEEE Software, 35*(1), 109-111. doi:10.1109/MS.2017.4541042

Rahman, A., Mahdavi-Hezaveh, R., & Williams, L. (2019). A systematic mapping study of infrastructure as code research. *Information and Software Technology, 108*, 65-77. doi:10.1016/j.infsof.2018.12.004

Rahman, A., & Williams, L. (2019). Source code properties of defective infrastructure as code scripts. *Information and Software Technology, 112*, 148-163. doi:10.1016/j.infsof.2019.04.013

Sanchez-Reillo, R. (2016). Signature analysis in the context of mobile devices. *Image and Vision Computing, 55*, 34-37. doi:10.1016/j.imavis.2016.03.011

Schermann, G., Cito, J., Leitner, P., Zdun, U., & Gall, H. C. (2018). We're doing it live: A multi-method empirical study on continuous experimentation. *Information and Software Technology, 99*, 41-57. doi:10.1016/j.infsof.2018.02.010

Schultz, D., Skarlupka, H., Brik, V., & Merino, G. (2019). CVMFS: Stratum 0 in Kubernetes. *EPJ Web of Conferences, 214*, 7032. doi:10.1051/epjconf/201921407032

Shore, J. J. M. (2015). An Obligation to Act: Holding Government Accountable for Critical Infrastructure Cyber Security. *International Journal of Intelligence and CounterIntelligence, 28*(2), 236-251. doi:10.1080/08850607.2014.962356

Tilkov, S. (2015). The Modern Cloud-Based Platform. *IEEE Software, 32*(2), 116-116. doi:10.1109/MS.2015.51

Xu, Y., Koren, I., & Krishna, C. (2017). AdaFT: A Framework for Adaptive Fault Tolerance for Cyber-Physical Systems. *ACM Transactions on Embedded Computing Systems (TECS), 16*(3), 1-25. doi:10.1145/2980763

Yilmaz, M., Tasel, F. S., Gulec, U., & Sopaoglu, U. (2018). Towards a process management life-cycle model for graduation projects in computer engineering. *PLOS ONE, 13*(11), e0208012. doi:10.1371/journal.pone.0208012

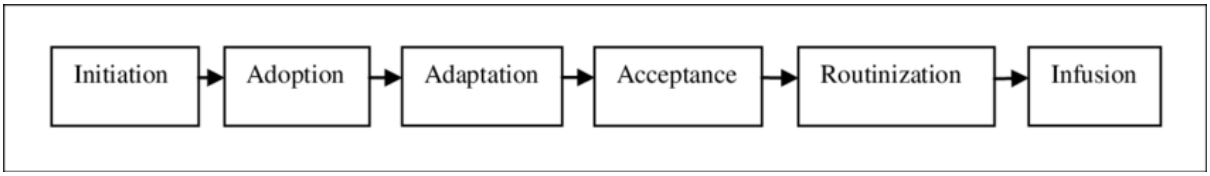
Younas, M., Jawawi, D. N. A., Ghani, I., Fries, T., & Kazmi, R. (2018). Agile development in the cloud computing environment: A systematic review. *Information and Software Technology, 103*, 142-158. doi:10.1016/j.infsof.2018.06.014

Zhu, L., Bass, L., & Champlin-Scharff, G. (2016). DevOps and Its Practices. *IEEE Software, 33*(3), 32-34. doi:10.1109/MS.2016.81

Zhu, L., Xu, D., Tran, A. B., Xu, X., Bass, L., Weber, I., & Dwarakanathan, S. (2015). Achieving Reliable High-Frequency Releases in Cloud Environments. *IEEE Software, 32*(2), 73-80. doi:10.1109/MS.2015.23

Bijlage 3: Informatie beschikbaar bij de interviews

IT Implementatieproces



Fase	Proces	Product
Initiatie	Actief en/of passief scannen van organisatorische problemen/kansen en IT oplossingen wordt ondernomen. Druk om te veranderen ontwikkelt zich vanuit organisatorische behoefte (pull), technologische innovatie (push) of beide.	Een overeenkomst is gevonden tussen IT oplossing en de toepassing daarvan in de organisatie.
Adoptie	Rationele en politieke onderhandelingen volgen om organisatorische steun te krijgen voor de implementatie van de IT toepassing.	Een beslissing is gemaakt om de vereiste middelen te investeren teneinde de implementatie mogelijk te maken.
Adaptatie	De IT toepassing wordt ontwikkeld, geïnstalleerd en onderhouden. Organisatorische procedures worden herzien en ontwikkeld. Mensen binnen de organisatie worden getraind in zowel de nieuwe procedures als de IT toepassing.	De IT toepassing is klaar voor gebruik in de organisatie.
Acceptatie	Mensen in de organisatie worden ertoe bewogen zich te committeren aan het gebruik van de IT toepassing.	De IT toepassing is in gebruik genomen bij het werk binnen de organisatie.
Routinisering	Gebruik van de IT toepassing als normale gang van zaken wordt aangemoedigd.	De IT toepassing wordt niet meer als iets bijzonders gezien.
Infusie	Toegenomen organisatorische effectiviteit wordt bereikt, door de IT toepassing op een meer uitgebreide en geïntegreerde manier te gebruiken, om aspecten van een hoger niveau van organisatorisch werk te ondersteunen.	De IT toepassing wordt in zijn volledige potentieel binnen de organisatie gebruikt.

Voordelen van CDV

Voordeel	Beschrijving
1. Kortere time-to-market	Een direct gevolg van snellere en meer frequente releases, is dat software sneller beschikbaar is op de markt.
2. Continue feedback	Door nieuwe software vaker en sneller uit te rollen naar de klant, wordt deze in staat gesteld vaker feedback te geven.
3. Verbeterde betrouwbaarheid en kwaliteit van de release	Het automatiseren van het testwerk en het uitrollen van nieuwe versies, zorgt ervoor dat de betrouwbaarheid en kwaliteit omhoog gaat.
4. Verhoogde klanttevredenheid	Klanten kunnen vaker en sneller feedback geven en deze feedback lijdt tot oplossingen en verbeteringen.
5. Verbeterde ontwikkelingsproductiviteit	Doordat een ontwikkelaar een nieuwe verbetering na ontwikkeling direct beschikbaar kan maken en dit kan terugdraaien als dat nodig is.
6. Snelle innovatie	Snellere feedback en beter contact met klanten leidt tot snellere innovatie.
7. Smallere testfocus	Een nieuwe versie van de software bevat minimale wijzigingen in de code.

Componenten van CDV

Component	Beschrijving
1. Snelle en frequente release	Het vermogen om software uit te rollen naar productie wanneer de organisatie wil, gebaseerd op de behoefte, maar gericht op een zo kort mogelijk cyclus (wekelijks of dagelijks), of zelfs continu.
2. Flexibel productontwerp en architectuur	CDV/CDP vereist een robuuste software architectuur met als doel een balans te vinden in snelheid en stabiliteit.
3. Continuous testing en quality assurance	De kwaliteit van de software is te allen tijde compromisloos verzekerd, ondanks snel en continu uitrollen naar ander omgevingen
4. Automatisering	Het automatiseren van de delivery pipeline, van initiële software build, tot testen, verder uitrollen en monitoren.
5. Configuratiemanagement	Versiebeheer van alle code en configuraties.
6. Betrokken klant	Mechanismen om klanten betrokken te krijgen bij het ontwikkelproces en zodoende zo vroeg mogelijk feedback te krijgen, teneinde invloed uit te oefenen op ontwerp en innovatie.
7. Continue en snelle uitvoering van experimenten	Het systematisch ontwerpen en uitvoeren van kleine experimenten die vervolgens de productontwikkeling gidsen en innovatie versnellen.
8. Post-deployment activiteiten	Activiteiten die worden uitgevoerd nadat het product (of uitbreiding of verbetering ervan) uitgerold is, om snelle beslissingsvorming te ondersteunen.
9. Agile en Lean	Verder uitbreiding op Agile en Lean softwareontwikkeling.
10. Organisatorische factoren	Organisatorische factoren die CDV/CDP mogelijk maken.

Determinanten

Determinant	Beschrijving
Klant	Een hoge frequentie van nieuwe softwareversies is voor de klant niet altijd gewenst. De aard van het (software)product en bijbehorende doelgroep dienen zorgvuldig in kaart te worden gebracht, alvorens CDV geïmplementeerd wordt en bij het bepalen in welke mate CDV geïmplementeerd wordt.
Domein	Het applicatiedomein heeft invloed op de manier waarop de software uitgerold kan worden naar een volgende omgeving en uiteindelijk de klant. Dit kan beperkingen geven aan de mate waarin CDV geïmplementeerd kan worden en de voordelen tot uiting kunnen komen. Derhalve dient het applicatiedomein grondig geanalyseerd te worden, alvorens CDV geïmplementeerd wordt.
Architectuur	Een geschikte architectuur is een zeer kritieke factor bij het realiseren van CDV. De software dient geautomatiseerd getest en uitgerold (deploy) te worden en dat kan alleen wanneer de architectuur van de software zich daarvoor leent. Een voor CDV geschikte architectuur wordt gevormd door de volgende aspecten: • Deployability - uitrolbaarheid: software dient te bestaan uit componenten die (betrouwbaar en gemakkelijk) los van elkaar uit te rollen zijn. • Modifiability - aanpasbaarheid: software dient gemakkelijk en vaak aangepast te kunnen worden, zonder dat dit een negatief effect heeft op ander software componenten. • Loggability and monitorability - vermogen om logging en monitoring toe te passen: bij het continu ontwikkelen, testen en uitrollen van software gaan dingen fout. Deze fouten dienen snel opgespoord en hersteld te worden, wat veel logging vereist. Deze logdata dient op een slimme manier gemonitord te worden. • Testability – testbaarheid: software dient snel en geautomatiseerd getest te kunnen worden. • Resilience – veerkracht: ga er vanuit dat componenten weleens kapot gaan. Het omvallen van een bepaald softwarecomponent dient dan minimale gevolgen te hebben. • Reusability – herbruikbaarheid: inzetten op herbruikbaarheid heeft een negatief effect op CDV, omdat onderlinge afhankelijkheden tussen componenten worden gecreëerd. Bij een ongeschikte architectuur, vaak te vinden bij oudere, monolithische software, zal sprake zijn van een langdurig en kostbaar verandertraject om de software -onder architectuur- opnieuw te bouwen in componenten.
Cultuur	CDV is een fenomeen in de lijn van Agile en Lean softwaredevelopment. Het vereist een cultuur van continu verbeteren, waarin nieuwe ideeën enthousiast worden omarmd, fouten worden gezien als kansen om van te leren, professionals ruimte en vertrouwen krijgen zodat zij sterk gemotiveerd blijven. Dit leidt tot het bewustzijn en de transparantie die vereist zijn om CDV tot een succes te maken.
Testen	De testinspanning van de organisatie zal zeer sterk toenemen en dient niet onderschat te worden. Handmatige tests en langdurige tests vormen een risico op het succes van CDV en dienen tot een minimum beperkt te worden.

Bijlage 4: Interviewschema

Thema: Introductie

Vraag	**Verduidelijking**
Wat is uw naam?	
Welke functie en/of rol bekleedt u?	Voor zover relevant. Een kandidaat kan bijvoorbeeld teammanager zijn en tevens de rol van product owner vervullen.
Hoelang bent u in dienst bij de organisatie en in deze functie/rol?	En in huidige functie?
Organisatie	Naam, omvang in aantal medewerkers, branche. Waar mogelijk zelf voorbereiden en bevestigen tijdens interview.

Thema: inzicht in situatie

Vraag	**Verduidelijking**
Wat is de status van CDV? Reeds geïmplementeerd, implementerend, of besloten tot implementatie?	Eventueel refereren aan IT implementatieproces. Onderscheid tussen eind fase 1 of eind fase 2. Onderscheid in de verdere fasen is voor het onderzoek niet relevant.
Hoe lang wordt al conform CDV gewerkt?	
Wordt het proces van softwareontwikkeling volledig door de organisatie zelf uitgevoerd?	Hangt mogelijk samen met het applicatiedomein
Hoe groot is het team dat software ontwikkelt op de wijze van CDV?	
Hoe vaak worden nieuwe softwareversies uitgerold?	(release)

Thema: Adoptiebeslissing
Begeleiden met IT implementatieproces en benadrukken dat de vragen betrekking hebben op de fase (periode) waarin de adoptiebeslissing is genomen. Daarbij: adoptie≠implementatie

Vraag	**Verduidelijking**
Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?	
Waarom was … daarbij belangrijk? Hoe is …. dan precies gegaan? Enz.	Doorvragen n.a.v. antwoord

Thema: Voordelen van CDV

Vraag	**Verduidelijking**
Welke doelen streeft de organisatie na met de toepassing van CDV?	Evt. doorvragen op huidige doelen en doelen bij adoptiebeslissing/aanvang CDV werkwijze en hun onderlinge verschillen.
Begeleiden met overzicht voordelen van CDV en opmerken dat deze voordelen in de literatuur genoemd worden.	
Welke van deze voordelen spelen zijn relevant of van toepassing voor uw organisatie?	Doelen≠voordelen, maar de voordelen kunnen organisaties in staat stellen bepaalde doelen te behalen.

Op welke manier speelden deze voordelen en/of doelen een rol bij het maken van de adoptiebeslissing?	
Was dat met de kennis van nu mogelijk anders gegaan?	Evt. doorvragen n.a.v. eerder antwoord
Had uw organisatie graag over dergelijk overzicht van CDV voordelen willen beschikken bij het maken van de adoptiebeslissing?	Evt. doorvragen n.a.v. eerder antwoord

Thema: Componenten van CDV

Begeleiden met overzicht van componenten van CDV en opmerken dat deze componenten in de literatuur genoemd worden.

Vraag	**Verduidelijking**
Wat zijn voor uw organisatie de belangrijkste componenten van CDV uit dit overzicht?	
Op welke manier hebben deze componenten een rol gespeeld bij het maken van de adoptiebeslissing?	
Was dat met de kennis van nu mogelijk anders gegaan?	Evt. doorvragen n.a.v. eerder antwoord
Had uw organisatie graag over dergelijk overzicht van CDV componenten willen beschikken bij het maken van de adoptiebeslissing?	Evt. doorvragen n.a.v. eerder antwoord

Thema: Determinanten

Begeleiden met overzicht van determinanten (die een rol kunnen spelen bij adoptiefase) en opmerken dat deze determinanten geïnterpreteerd zijn uit de literatuur.

Vraag	**Verduidelijking**
Welke invloed heeft de factor klant gehad bij het maken van de adoptiebeslissing?	Wordt software ontwikkeld om direct klanten te bedienen of betreft het de ondersteuning van interne processen? Of beide?
	Doorvragen n.a.v. antwoord
Zijn er applicaties of onderdelen van de informatievoorziening waarbij is besloten om niet tot de CDV werkwijze over te gaan vanwege de factor klant?	
Welke invloed heeft de factor applicatiedomein gehad bij het maken van de adoptiebeslissing?	Voorbeelden applicatiedomein: embedded software, mobile apps,
	Doorvragen n.a.v. antwoord
Zijn er applicaties of onderdelen van de informatievoorziening waarbij is besloten om niet tot de CDV werkwijze over te gaan vanwege de factor applicatiedomein?	

Welke invloed heeft de factor architectuur gehad bij het maken van de adoptiebeslissing?	Ter ondersteuning kunnen de verschillende aspecten van een geschikte architectuur genoemd worden.
	Doorvragen n.a.v. antwoord
Zijn er applicaties of onderdelen van de informatievoorziening waarbij is besloten om niet tot de CDV werkwijze over te gaan vanwege de factor architectuur?	
Welke invloed heeft de factor cultuur gehad bij het maken van de adoptiebeslissing?	Het gaat hier om de organisatiecultuur. Evt. uitleggen met de definitie die Humble geeft: “a pattern of shared tacit assumptions that was learned by a group as it solved its problems of external adaptation and internal integration, that has worked well enough to be considered valid and, therefore, to be taught to new members as the correct way to perceive, think, and feel in relation to those problems.”
	Doorvragen n.a.v. antwoord
Zijn er applicaties of onderdelen van de informatievoorziening waarbij is besloten om niet tot de CDV werkwijze over te gaan vanwege de factor cultuur?	
Welke invloed heeft de factor testen gehad bij het maken van de adoptiebeslissing?	
	Doorvragen n.a.v. antwoord
Zijn er applicaties of onderdelen van de informatievoorziening waarbij is besloten om niet tot de CDV werkwijze over te gaan vanwege de factor testen?	
Welke eventuele andere factoren hebben invloed gehad bij het maken van de adoptiebeslissing? Wat kunt u hierover vertellen?	Deze vraag komt op hetzelfde neer als de eerste vraag bij het thema ‘adoptiebeslissing’, maar wordt concreter gesteld.
Terugkijkend op de gemaakte adoptiebeslissing en het vervolg van de implementatie van CDV, hadden de opgestelde determinanten een dusdanige rol kunnen spelen dat de adoptiefase beter of sneller was doorlopen? Met andere woorden, had u de determinanten destijds graag willen weten?	

Bijlage 5: Transcriptsamenvattingen

Interview 1

Functie, periode	Enterprise architect, voor een half jaar. Nu drie jaar werkzaam voor de organisatie in vergelijkbare functies.
Organisatie	Retail, circa 750 fysieke winkels wereldwijd en online winkel. 700 medewerkers in ondersteunende diensten, waarvan 70 in de IT.
Status CDV	Onderweg van CI naar CDV Context: verschilt per team. Een bepaald team loopt erg voorop, een team moet eigenlijk nog beginnen en het laatste team bungelt daar tussenin. Dit heeft te maken met de volwassenheid in de relatie met de leverancier en ook de tooling die gebruikt wordt, die is niet erg geschikt om snel een CDV werkwijze mee te starten. Het is nog geen grote strategische beslissing geweest op organisatieniveau.
Softwareontwikkelproces	De organisatie huurt externe personeel in om te helpen bij de softwareontwikkeling. Daarnaast zijn delen van de softwareontwikkeling ook uitbesteed.
Releasefrequentie	Gemiddeld iedere 2 weken, sommige teams vaker.
Scope	De organisatie werkt met verschillende teams over verschillende domeinen. De status van CDV verschilt daarbij ook. De scope voor het gesprek is de gehele organisatie.

Status CDV:
Wij hebben meerdere softwareontwikkelteams, vaak extern ingehuurd, die ieder verantwoordelijk zijn voor een bepaald domein of verschillende services. Deze teams werken onafhankelijk van elkaar. Uiteraard moeten er bepaalde interfaces beschikbaar zijn, maar op zich zijn ze onafhankelijk. Ze hebben hun eigen omgeving en manier van werken. De meeste teams werken met CI/CD, maar sommigen werken er überhaupt niet mee. **De status van CDV verschilt per team. Het staat de teams min of meer vrij om hun eigen werkwijze toe te passen**. Maar we willen er wel voor zorgen dat alles steeds beter en sneller werkt. **Het doel van de organisatie is om zoveel mogelijk waarde uit de kosten voor deze teams te halen. CDV is een middel om meer Agile te werken** en dat willen wij wel. Het idee is dus wel om CDV overal mogelijk te maken. Maar soms zijn we gewoon nog niet zover. **En dat komt omdat mensen de kennis of de vaardigheden nog niet hebben, of omdat de omgeving waarin zij werken, bijvoorbeeld SAP, dit nog niet mogelijk maakt.** Dus samenvattend vind ik dat onze organisatie in het midden uitkomt. We zijn ermee bezig, voor sommige teams is het al goed gelukt, voor andere teams zijn we nog niet zover en moeten we goed uitzoeken hoe we het daar ook kunnen uitrollen. Voor deze materie geldt ook dat het eigenlijk nooit af is. Je kunt altijd verder uitbouwen, optimaliseren en automatiseren. Maar wij zijn in onderdelen wel zo ver dat een verandering in de code, een code commit, leidt tot een automatische deployment op een testomgeving of volgende stage volgt. Daar zijn wij dus heel dichtbij het ideaalplaatje. Het enige wat we niet doen is dat teams wijzigingen automatisch naar productie brengen. Productiedeployments zijn bij ons handmatig getriggerd, maar ook daarvoor geldt dan dat het met een druk op de knop in productie staat. Ook dat hebben we geautomatiseerd.

Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?
Je zit fout door te denken dat een organisatie dit bewust introduceert. Daar zijn wij denk ik niet. Ik kan een beetje verklaren hoe het bij ons is gegaan. Zo'n 2 tot 3 jaar geleden waren wij een heel traag

IT bedrijf. Alles was uitbesteed aan één partij. Alles werd conform watervalmethode ontwikkeld. Een jaar bouwen, half jaar testen en dan uitgerold worden. Alles was supertraag. Vervolgens is er een nieuwe e-commerce stack geïntroduceerd en dat was eigenlijk het begin van een bedrijfstransformatie. Er zijn 3 architecten met technische kennis, waaronder ik. **We zijn bezig om een omgeving te creëren waarin we efficiënt kunnen werken.** We voorzien in een ondersteunende behoefte. Het gaat erom dat we de business en in relatie daarmee, PO's en ontwikkelteams ondersteunen. We delen kennis en vaardigheden om ervoor te zorgen dat de teams efficiënter werken. In later stadium zouden we ook minimumeisen kunnen stellen en zouden we CDV kunnen introduceren. Daar zijn we nu nog niet mee bezig. Ik zou zeggen dat we nog in een stadium vóór dit proces zitten. **We denken niet actief na over organisatorische kansen en problemen waarbij CDV de oplossing is.** We willen uiteindelijk wel van een ondersteunende rol naar een meer directieve rol. Dan zouden we terecht komen in dit IT implementatieproces. We zijn echter wel met CDV bezig. Er zijn teams binnen onze organisatie die software automatisch deployen, zelfs naar productie. **Dat doen zij van zichzelf uit en dat kunnen ook partijen uit de markt zijn.** Het proces start normaliter met een businesswens voor een nieuwe applicatie. Vervolgens kijken we of we al over een team of domein beschikken waarbinnen de wens vervuld kan worden of dat we een nieuw team nodig hebben. Als we een nieuw team nodig hebben, dan zoeken we op de markt naar een geschikt team. **Uiteraard zoeken we dan naar een team dat op een moderne manier software ontwikkelt**. **Maar er zijn meerdere criteria, zoals bepaalde domeinkennis. Een specialist in mobile apps kun je niet gebruiken voor replenishment. Soms moet je bij je keuze compromissen maken. We moeten soms contracten aangaan met teams die niet op een absoluut moderne manier werken, maar die wel de absolute expert zijn in hun domein.** Replenishment is een voorbeeld van een beetje een ouderwets domein. Je vindt daar heel weinig moderne partijen. **Op dit moment lijkt overgaan tot de werkwijze van CDV daarom meer op iets waar je inrolt, dan een expliciete keuze. Wij proberen het met onze huidige middelen generiek naar een moderne manier van werken te sturen, maar we hebben het niet als doel op zich gesteld.** Wij hebben bepaalde non-functional requirements opgesteld waar nieuwe teams en applicaties zoveel mogelijk aan moeten voldoen. Dit teneinde API-gedreven te gaan werken. Daarbij vragen we ook expliciet of deployments automatisch gebeuren en tests ook geautomatiseerd zijn. Maar we hebben CDV op zich niet als non-functional requirement gesteld, omdat we bang zijn dat we dan een heleboel teams of bedrijven uitsluiten. **Retail kent deels echt nog een erg traditionele manier van softwareontwikkeling. Sommige onderdelen zoals het hele kassa-applicatielandschap, datawarehouselandschap, de hele replenishment, alles wat met logistiek heeft te doen, dat is zo ouderwets dat het niet mogelijk is om CDV toe te passen**. Bijna al deze software is lang geleden ontwikkeld. Dat is vaak zware, monolithische software met veel afhankelijkheden. **En de mensen spelen hierbij ook deels een rol. Die hebben ergens in de jaren '90 de software gebouwd, dus daarmee hebben ze een oude manier van denken.** Daarnaast denk ik dat een **architectuur geschikt moet zijn om CDV toe te passen**. **En veel van deze applicaties volgen deze basisprincipes niet, dus die zijn technisch niet in staat het te doen. Als je een grote monoliet hebt, is het vaak niet mogelijk geautomatiseerd te testen.**

<u>Doelen/voordelen:</u>

Op businessniveau is efficiënter werken het doel. Daaronder zou je nog subdoelen kunnen zien. Bijvoorbeeld jonge en goede mensen die van de universiteit komen, daar iets hebben geleerd en niet in een omgeving terecht willen komen waar hun geleerde principes niet worden nageleefd. **Je wil een uitdagende werkomgeving voor nieuwe en jonge medewerkers. Omdat wij denken met deze mensen een meer flexibele en Agile manier van werken gedaan te krijgen en onszelf te blijven uitdagen.** Daarnaast willen we **sneller kunnen reageren** wanneer er iets misgaat. Ook willen we

complexere releases vermijden, door heel veel kleine release te doen. Maar dat zie ik echt als **tactische doelen, onder het grotere geheel.**

Overzicht voordelen CDV:

De opsomming van de voordelen zou op zich zeker kunnen helpen. Echter, ik zie de uitdaging in het introduceren van CDV niet zozeer in het overtuigen van mensen dat het zinvol is. Ik zie het meer in het technisch mogelijk maken. Een bank bouwt bijvoorbeeld veel software zelf. Wij als retailer doen dat niet. Dat komt omdat er veel meer software op de markt beschikbaar is en we redelijk gestandaardiseerde processen hebben. **Wij zijn dus meer bezig met het integreren van standaardsoftware dan het bouwen van software**. Daardoor hebben we wel te maken met de beperkingen van deze software. **En in de onderdelen waar wij geen technische architectuur beperkingen hebben, daar doen wij het al.**

Componenten:

Ik denk eigenlijk dat je dit allemaal nodig hebt. Het is allemaal noodzakelijk. Voor onze situatie geldt dat voor sommige softwarecomponenten nummer 2 niet is gegeven. In sommige onderdelen wil onze business afdeling gewoon niet betrokken zijn. Daar hebben wij nog veel werk te verzetten. **Dit overzicht laat eigenlijk heel goed zien waar wij mee zitten**. **Wij zitten met noodzakelijkheden en beperkingen buiten onze context waardoor sommige van deze componenten ons niet gegeven zijn en wij CDV dus niet kunnen inzetten.**

Determinanten:

Je kunt hier niet alle domeinen hetzelfde in behandelen. Onze klanten bij e-commerce willen dit heel graag. Andere afdelingen willen het niet. Bij e-commerce domein is er ook nog eens erg geschikt voor. **Ons replenishment domein is er gewoon niet geschikt voor**. **Wij hebben waarschijnlijk een stuk 20 domeinen, maar helaas hebben wij geen adequaat overzicht van al onze domeinen. Grofweg zie je dat de front-end applicaties, denk bijvoorbeeld aan webapplicaties met een focus op business to consumer, erg modern zijn. Daarnaast zie je dat voor alle back-end applicaties zoals SAP of een datawarehouse, onze klant dingen eigenlijk helemaal niet snel wil veranderen**. Die willen eigenlijk dat alles stabiel blijft. **Dat is niet alleen bij de klant, maar ook bij de leverancier zo. Cultuur speelt hierbij absoluut een rol. Je ziet daar vaak culturen van mensen die het al 20 jaar doen en die niet zo houden van verandering. Omdat ze in het verleden al meegemaakt hebben dat veranderingen slecht voor hen waren.**

De onderwerpen die staan benoemd bij architectuur, die zijn grote monolieten doorgaans niet goed gegeven. Een SAP deployment is vaak uren werk. Aanpasbaarheid is niet goed gegeven, omdat alles van alles afhankelijk is. Testability is ook niet goed gegeven, omdat je componenten niet los kunt aanspreken. Resilience is er ook niet echt. Als er iets kapot gaat, gaat alles kapot.

Het front-end domein is over het algemeen gewoon nieuwer. Webapplicaties zijn deze eeuw opgekomen, daarvoor bestonden ze eigenlijk niet. Je ziet dat vaker. **Bij domeinen die nog niet zo lang bestaan zie je dat ze moderner ingericht zijn.**

Voor alle determinanten geldt dat ze weleens de stap naar CDV in de weg zitten. **Architectuur en domein zijn sterk aan elkaar gekoppeld en zijn voor ons de belangrijkste determinanten**. Onze interne klanten kunnen wij veranderen, de cultuur van teams kunnen wij veranderen door een ander team te kiezen. Op testinspanning hebben we zelf grip. Waar we geen grip op hebben is beschikbare software in de markt en daarmee op domein en architectuur. Architectuur hebben we enigszins grip op, maar dan op hoger niveau, de integratie. Niet op de onderliggende softwarearchitectuur.

Deze determinanten helpen mij te beschrijven waar we staan, ik mis er geen.

Interview 2

Functie, periode	Senior Product Owner voor de laatste twee jaar. En vier jaar bij deze organisatie als Product Owner.
Organisatie	Retail, circa 750 fysieke winkels wereldwijd en online winkel. 700 medewerkers in ondersteunende diensten.
Status CDV	Onderweg van CI naar CDV Context: verschilt per team. Een bepaald team loopt erg voorop, een team moet eigenlijk nog beginnen en het laatste team bungelt daar tussenin. Dit heeft te maken met de volwassenheid in de relatie met de leverancier en ook de tooling die gebruikt wordt, die is niet erg geschikt om snel een CDV werkwijze mee te starten. Het is nog geen grote strategische beslissing geweest op organisatieniveau.
Softwareontwikkelproces	De organisatie huurt een team ontwikkelaars in van een externe leverancier
Releasefrequentie	Minimaal iedere 2 weken
Scope	De organisatie werkt met verschillende teams over verschillende domeinen. De status van CDV verschilt daarbij ook. De scope voor het gesprek is het team (domein) online front-end (3 teams).

Status CDV:
Dit verschilt per team. Een team is duidelijk het meest gevorderd daarin. Een ander team moet er eigenlijk nog mee beginnen en het laatste team bungelt daar een beetje tussenin. **Dat komt aan de ene kant door de volwassenheid van de relatie met de leverancier en aan de andere kant de tooling die zich er niet zo goed voor leent om snel zoiets op te starten.** En in het algemeen is het nog niet een groot strategisch punt geweest voor de gehele organisatie. De werkwijze wordt dus niet afgedwongen. De IT afdeling in de huidige vorm bestaat nu een jaar. Een afdeling daarbinnen houdt zich bezig met architectuur, infrastructuur en quality assurance, maar die zijn nog met vrij basale dingen bezig. Wellicht dat we van hen in later stadium onze werkwijze moeten aanpassen. Concluderend kunnen we zeggen dat we van CI onderweg zijn naar CDV.

Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?
ja, nou vanuit de web kant kan ik daar wel iets over vertellen. Toen ik binnenkwam bij de organisatie, toen kwam ik in een situatie terecht waarin de website volgens mij drie keer per jaar gereleast werd. en dat was vrij vrij heftig. Dat betekent dat je dat er echt **een week lang doortesten was voor elke release. En dat heel veel mensen dat moesten doen, waar het echt veel nachtwerk ook werd**. En tegelijkertijd was dat natuurlijk in een periode waarin de organisatie eigenlijk al verder zou moeten zijn dan dat, want je zag, en dan praat ik dus over 2016. **Er gebeurde zoveel online en ook zoveel klantvragen daarover waar we eigenlijk enerzijds niet in konden voldoen qua snelheid destijds**. En anderzijds, **Als we iets deden dan kostte dat heel veel geld en heel veel tijd.**
Dus toentertijd is besloten om, ook om **een andere mindset te geven aan heel veel mensen** binnen de organisatie om te starten met Scrum. En dat was eigenlijk allemaal gebaseerd op **externe factoren. In die zin, ik was nieuw, mijn manager was nieuw en er is een andere web agency aangehaakt, dus er kwam heel veel vers bloed** in om dat te gaan doen. Met uiteraard de belofte dat sommige dingen **conversieverhogend zouden werken en kosten zouden besparen. Dat neigt misschien naar een business case. Ik weet niet of er nog een hele zware business case lag voordat ik en mijn manager binnenkwamen**. Maar ik weet wel dat allles wat daarna gebeurd is, is gebaseerd op doelstellingen die getoetst kunnen worden door, **ja dingen sneller live te zetten.** De bestaande

releases waren namelijk heel duur. Vooral als je kijkt wat je ervoor opgeleverd krijgt. Juist omdat het zo'n zwaar proces is met, bij wijze van spreken een half jaar bouwen en dan mergen en dan testen, wat **dan de kwaliteit ook niet altijd ten goede kwam**. Wat ik vernam is dat het ook wel een **vrij prijzig geintje was voor datgene wat er in een release zit. Ten opzichte van de uitgave, wordt er weinig waarde opgeleverd.**

Doelen/voordelen:

In theorie zou ik zeggen alles een rol speelde, maar de praktijk is vaak weerbarstiger. Voor het maken van de beslissing zie ik 1 overduidelijk. We releasen nu minimaal iedere 2 weken. Nummer 2 komt nog niet echt van de grond. In de praktijk zijn er nog heel veel kansen. Nummer 3 is overduidelijk een keuze en voelen we ook als voordeel. **Verhoogde klanttevredenheid komt voort uit in staat zijn eerder iets te leveren wat de klant wil, of eerder iets te fixen waar de klant last van heeft.** Nummer 5 merken we wel een beetje. Ontwikkelaars herstellen fouten sneller omdat betreffende kennis nog vers is. We draaien nog weleens iets terug en dat is dan meer dan een enkele feature. Releases zijn nog iets groter, we releasen nog niet per feature. Snelle innovatie en smallere testfocus voeg ik even samen omdat dat in een apart traject aandacht krijgt. Het evidence based growth traject, waarbij we meer met A/B testing gaan doen. Dat zit nu nog niet zo in het ontwikkelproces verweven.

We waren ons ten tijde van de beslissing niet bewust van al deze voordelen. Je kiest een paar hoofdargumenten uit om mensen mee te krijgen om dit te gaan doen. Vervolgens ontstaat er een andere dynamiek en merk je dat ook de andere voordelen gaan opleveren of dat je ze kunt benoemen bij een vervolgstap. **We zijn niet begonnen met tien voordelen of iets dergelijks. Dat verkoopt zo slecht. Ik denk dat je een aantal voordelen moet uitkiezen die het meest tot de verbeelding spreken voor de organisatie.** Kortere time-to-market lijkt ook wel iets waar je eerder wat aan zal hebben dan bijvoorbeeld snelle innovatie. **Je moet argumenten uitkiezen die snel iets opleveren, dan kun je mensen meekrijgen.**

Componenten:

Nummer 1 uiteraard. Nummer 3. Nummer 9 is ook wel belangrijk geweest toentertijd, we gingen namelijk voor het eerst die kant op, waarbij ook naar DevOps werd gekeken. **Nummer 8 was ook nog wel relevant**, omdat dat heel heftig was voor die tijd. Het ondersteunen en gebruiken van nieuw releases kostte veel moeite.

Ja, ik denk dat dit overzicht zou kunnen helpen bij het maken van de adoptiebeslissing. Men brengt te gemakkelijk buzzwords naar voren. Even een stap terug zou dan terecht zijn, wat betekent dat precies en wat hebben we daarvoor nodig? En dat beschrijven deze componenten denk ik. Wij waren ons dan bijvoorbeeld bewuster geweest van automatisering en waren wellicht eerder aan de slag gegaan met experimenten.

Determinanten:

Ik denk dat het in het geval van testen voor ons andersom was. Juist omdat we tests hebben geautomatiseerd zijn de handmatige tests afgenomen en daarmee de testinspanning ook. Daarom is het voor ons niet zo'n belangrijke determinant.

Ik denk dat cultuur erg belangrijk is, dat werd bij ons gedreven door het idee dat er echt iets nieuws moest komen en er kwam ook veel nieuw bloed binnen toen. **De manier van werken had ook impact op de cultuur. Door iedere twee weken iets op te leveren namen de transparantie en het vertrouwen toe.** Dus het cultuurstuk vind ik erg belangrijk. **De rol bij de adoptiebeslissing vind ik lastig. Ik denk dat de verandering vooral beargumenteerd is op harde factoren, de voordelen zoals time-to-market. Maar daarachter zat veel meer.** De beweging die je hiermee op gang brengt

is erg belangrijk. **De verandering kwam juist ook doordat er relatief veel personeelswijzigingen waren in die tijd. Wellicht vond men dat nodig om een verandering in cultuur te bereiken. Ik denk dat het lastig is om oorzaak en gevolg daarbij aan te wijzen, maar het lijkt zeker een relatie te hebben.**
De klant was bij de beslissing niet zo belangrijk.
Online front-end is bij ons wel het makkelijkste domein om dit in gang te zetten. Ik denk dat domein meer een hygiënefactor is. **Als het er is dan is het oké en als het er niet is heb je een probleem. We waren ons daar niet heel bewust van.**
Voor architectuur zijn wel iets meer bewuste keuzes gemaakt. We hebben daar een aantal non-functional requirements opgesteld die wel van belang zijn. Daarin zie ik wel overlap met CDV. Bij architectuur moet je ook wel, omdat het bepalend is. We proberen zoveel mogelijk alles met API's en microservices te verbinden. De non-functionals zijn met zorg opgesteld, maar lijken niet per se voort te komen uit een CDV ambitie. **De eisen die de steeds veranderende wereld stelt komen terug in de non-functional requirements en zijn in lijn met CDV. Je wil zaken als loggability and monitorability gewoon hebben, ongeacht CDV. Maar ze helpen ook bij CDV.**
Het domein kan ook erg beperkend zijn. Een team van ons heeft daar erg mee te maken. We krijgen daar zelfs CI niet aan de praat. Dit komt door off-the-shelf software. Deze voldoet dus niet aan de non-functional requirements en er is dus een relatie met architectuur. We kunnen in dat domein niet zomaar code deployen naar productie. Er moet dan veel handmatig worden nagebouwd.
Geld is wel een factor. Het is niet zozeer een determinant denk, maar eerder een gevolg van deze determinanten. Wij hebben met veel dingen een frisse start gemaakt en daardoor hebben we een vrij moderne architectuur kunnen neerzetten. Maar als ik het bijvoorbeeld heb over het team dat met veel off-the-shelf software werkt, zij kunnen niet zo makkelijk op deze ontwikkeling meeliften. Als ik daar een slag zou willen slaan dan kom ik op kosten uit die ik niet direct kan verantwoorden met de waarde die dat team oplevert. Ik krijg binnen dat domein de business case niet rond. Ik zie dat als een gevolg van de factor domein, maar het speelt een erg belangrijke rol is wat ik voor elkaar kan krijgen.
Wat ik nog een beetje mis zit in de organisatorische factoren. Business en IT moeten veel dichter bij elkaar komen. Het team dat software oplevert moet ook mensen uit de business bevatten.

Interview 3

Functie, periode	Enterprise architect, bijna twee jaar, in totaal veertien jaar werkzaam bij de organisatie in diverse functies
Organisatie	Schadeverzekeraar, tot 500 medewerkers
Status CDV	De organisatie beschikt over een ontwikkelstraat waarin CDV volledig omarmt is. Voor andere onderdelen geldt dat niet.
Softwareontwikkelproces	Uitsluitend met eigen mensen. Team van momenteel 12 medewerkers.
Releasefrequentie	Varieert sterk. Tien keer per dag komt voor, nul keer ook. Gemiddeld wordt elke dag wel iets naar Productie gebracht.
Scope	Java domein (batch- en webapplicaties)

Status CDV:

Wij hebben ook domeinen met een lagere releasefrequentie, maar hebben het zo ingericht dat ons Java domein onafhankelijk van andere domeinen van releasen. Anders zou je geen CI/CD kunnen toepassen vanwege de afhankelijkheid. **We hebben dat middels loose coupling bijna volledig kunnen elimineren.**

Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?

Dit is vanuit IT bottom-up gepusht. IT vond dat het beter kon. We hadden integratieproblemen en erg veel last van handmatige handelingen. We zitten op een ontwikkelstack waarbij instellingen steeds handmatig op iedere omgeving opnieuw gedaan moesten worden. **Door de menselijke handelingen was sprake van een hoge foutgevoeligheid en de business was daar eigenlijk wel klaar met al die fouten. Hierover werd over geklaagd door de business, ook omdat het zoeken en herstellen van fouten lang duurde. Eigenlijk duurde het allemaal te lang en was het onvoorspelbaar.** We waren niet in staat wijzigingen te versioneren en waren niet in control. **Vanuit IT is toen het initiatief gekomen om conform CI/CD te gaan werken, met de overtuiging dat deze investering wel terugverdiend zou worden. Er is toen budget beschikbaar gekomen om zelf een volledige CI/CD straat te ontwikkelen.**

Bij de implementatie is sprake geweest van een tweetrapsraket. We hebben eerst het credo opgevat 'script everything, version everything'**.** Dat zijn eigenlijk de twee pijlers geweest waarmee we de basis hebben gelegd. We hebben dus niet in eerste instantie CI/CD gebouwd. Automatisch deployen zou later komen. Eerst alles scripten, geen handmatige handelingen meer. Pas als je alles gescript hebt kun je nadenken over testautomatisering of CI/CD. Versioneren helpt bij het automatisch kunnen terugrollen. **Toen we die twee zaken hadden gedaan, gingen we kijken naar CI en CDV. Voor CDV hebben we besloten een deploymentstraat in te richten voor de hele OTAP straat.** Dat was stap twee. Hierbij was het doel om met een klik op een knop een applicatie kunnen inrichten op een willekeurige omgeving, dat die omgeving wordt opgespind, dat automatisch alle configuratie-items worden ingesteld en dat automatisch de applicatie wordt geïnstalleerd. Het is daarmee even gemakkelijke om een roll-forward te doen als een roll-back. **Stap drie was testautomatisering. We hebben daarbij verschillende, ook functionele tests, geautomatiseerd**. Wanneer een wijziging daar niet doorheen komt, mag hij niet naar productie. Hier was overigens ook CDP mogelijk, maar die status hebben we nooit bereikt. Dit komt omdat de business nooit genoeg vertrouwen heeft gekregen in het automatische testtraject. Ze wilden altijd nog zelf een controle doen.

Doelen/voordelen:

Nul op de meter, wat betekent geen fouten waar de klant iets van merkt. Wat daar achter zit is dat **wij vinden dat je als organisatie geen CDV kan doen zonder continuous testing.** Wat dus betekent dat ieder stukje code bij ons wordt getest, alvorens het wordt gedeployt.
Een andere is time-to-market verkorten. Dat is eigenlijk de belangrijkste. Dat we sneller in staat zijn dingen naar productie te brengen die belangrijk zijn voor het bedrijf.
Een derde is transparantie en traceerbaarheid. Als je beschikt over een geautomatiseerde lijn waarmee je continue software oplevert, is elke stap reproduceerbaar. Dus je kan aan iedereen aantonen hoe je development lifecycle verloopt. Als verzekeraar staan wij onder toezicht van de nodige instanties. Zij willen weten of we op een veilige en betrouwbare wijze ontwikkelen. **Het gaat uiteindelijk om de aantoonbaarheid van onze interne beheersing. Eigenlijk is dat het doel. Transparantie en traceerbaarheid helpen ons bij de aantoonbaarheid van onze interne beheersing.**

Overzicht voordelen CDV:
Time-to-market is helder. De nul op de meter komt terug in 3, 4 en 5. Doordat je minder tijd kwijt bent aan het herstellen van fouten wordt je ontwikkelingsproductiviteit namelijk verhoogd. **Eigenlijk hebben al deze voordelen meegespeeld en ik vind dat ze vaak ook erg in elkaars verlengde liggen. Ze hebben alle zeven meegespeeld bij de adoptiebeslissing. We hadden ook hulp van een externe partij. Zij hadden de voordelen vrij scherp aan ons gepresenteerd, dus ik herken eigenlijk volledig wat hier staat.**

Componenten:
De eerste is helder, die hangt samen met korte time-to-market. De tweede speelde niet echt een rol. Er was niet meteen behoefte aan flexibiliteit. Het reguliere releaseschema bleef gehandhaafd en nieuwe functionaliteit bleef nog weleens op acceptatie staan, ook al was het af. Het is goed dat hij er staat, want het klopt wel, maar bij ons speelde het geen rol. Continuous testing speelde in het begin ook niet mee, nu is dat wel heel belangrijk. Automatisering was wel belangrijk. We wilden handmatige handelingen elimineren. Daarnaast is het sneller en uiteindelijk ook goedkoper. De vijfde hangt daarmee samen. We hebben het configuratiemanagement geautomatiseerd. Script everything, version everything komt in deze punten terug. **De betrokken klant heeft eigenlijk geen rol gespeeld bij de keuze voor CI/CD. Dit is later gekomen bij de Agile transitie. Dat was namelijk geen technologische transformatie, maar een bedrijfstransformatie.** Zeven heeft voor ons geen rol gespeeld. Voor acht vind ik het lastig om te zeggen in hoeverre het een rol speelde bij de adoptiebeslissing. Het is helder dat het nu terugkomt in hoe we omgaan met roll-back versus roll-forward. Ik kan mij niet meer herinneren wanneer we een rollback hebben gedaan. Dat het iets veranderd heeft is evident, maar of we dat vooraf zagen aankomen kan ik niet zeggen. Agile en Lean speelden niet mee bij de keuze. **Organisatorische factoren: Wat we in de organisatie zagen is dat de bezetting van mensen hield de manier van werken in stand. Een release iedere drie maanden was prima voor de oude garde. De jongere collega's pikten dit niet en wilden verandering. Daar is de kentering langzaam uit ontstaan. Je zag de buitenwereld veranderen en de medewerkers verjongen. Ik denk dat dit een grote rol speelde.** De verschuiving van de verhouding tussen de twee groepen medewerkers zorgde er eigenlijk voor dat we nieuwe werkwijzen konden introduceren. **Ook de kredietcrisis heeft hier aan bijgedragen. Dat heeft zoveel littekens achtergelaten in de financiële sector, je moest wel naar de klant gaan luisteren. Klanten beïnvloeden organisaties namelijk.** Kijk bijvoorbeeld naar de groene revolutie. Organisaties veranderen, omdat klanten het afdwingen.
Dit overzicht van componenten had gesprekken destijds wel makkelijker gemaakt. We hebben wel externe hulp gehad, maar een concreet lijstje als deze heb ik nog nooit gezien. Ik had het zeker willen hebben.

Determinanten:

De rol die de klant speelde is dat we uiteindelijk alles doen voor de klant en bij alles de overtuiging hadden dat het ook echt beter zou worden voor de klant. Dat de klant het niet zou willen is eigenlijk nog nooit bij mij opgekomen. Dit komt natuurlijk ook omdat wij ons bezig houden met webapplicaties. Nieuwe releases uitrollen kan zonder dat de klant daar enige last van ondervindt. Voor domeinen buiten dat van mij zie ik wel dat deze factor bepalend kan zijn. Voor onze mainframe systemen doen we het niet. We krijgen testautomatisering niet aan de praat, het is een grote ramp. Voor onze AS400 systemen doen we het niet. We hebben testautomatisering redelijk aan de praat gekregen, maar het is de investering gewoon niet waard daar. Het is echt lastig. We zijn daarnaast ook bezig het inrichten van een nieuw systeem, Axon van Keylane. Het is een modern systeem, maar zelfs daar zien we dat CI/CD niet zo makkelijk gaat. Testautomatisering moeten we buiten de pipelines doen. Dus daar lukt het niet zo goed. Voor Salesforce krijgen we het bijvoorbeeld weer voor elkaar. **Dus het is echt de vraag of de techniek die je gebruikt, de visie of werkwijze ondersteunt.**

En daarmee kom je eigenlijk uit op architectuur. De link die je legt tussen domein en architectuur is eigenlijk wel terecht. Voor ons Java domein geldt dat we middels loose coupling CDV mogelijk konden maken. De eisen die in dit overzicht gesteld worden aan de architectuur hebben we naderhand binnen ons domein kunnen realiseren. Ten tijde van de beslissing waren we ons hier niet zo bewust van. De Java ontwikkelstack hadden we namelijk al ruim voordat we besloten conform CI/CD te gaan werken. In zekere zin is er misschien wel sprake van toeval. Al weet niet wat er was gebeurd als onze ontwikkelstack ongeschikt was gebleken. Mogelijk is architectuur gewoon een terechte determinant, maar viel dat in ons geval niet zo op, omdat het op orde was. Cultuur heeft een heel grote rol gespeeld. Het was een grote investering van de business om zoiets op te gaan zetten. Daarbij is steeds vertrouwen geweest in de professionals en het feit dat het ook iets gaat opleveren. De cultuur van continue verbeteren die is hier gewoon aanwezig. In het begin is ook het nodige verkeerd gegaan. De hele boel heeft er weleens volledig uitgelegen. Zelfs toen bleef het vertrouwen, in het begin kan zoiets gebeuren werd gedacht. **Ik denk dat het niet mogelijk is CDV op te zetten zonder dat je bedrijfscultuur hebt van vernieuwing en vertrouwen.** Fouten worden weleens gemaakt en soms zit het even tegen. Als hier geen ruimte voor is, gaat het niet lukken. Dit is echt een vereiste. **Ik denk overigens niet dat iedereen zich daar bewust van was ten tijde van de beslissing. Sommigen wel en anderen zijn hier waarschijnlijk in gevolgd.**

CI/CD kan alleen als geautomatiseerd testen werkt. Het is een randvoorwaarde. We hebben de nodige eisen gesteld aan testen die we checken, alvorens een applicatie een CI/CD applicatie mag zijn. We beschikken feitelijk over een checklist: er moet versiebeheer zijn, er moet een pipeline zijn, er moet geautomatiseerde kwaliteitscontrole op de code zijn, er moeten geautomatiseerde tests zijn, de tests moeten allemaal atomair zijn, alles moet op dezelfde server staan.

Interview 4

Functie, periode	Product Owner, ruim anderhalf jaar
Organisatie	Online mediabedrijf, 65 medewerkers in dienst
Status CDV	Werkwijze geïmplementeerd, sinds 3 maanden
Softwareontwikkelproces	Eigen team software development, ca. 12 medewerkers
Releasefrequentie	Sinds CDV meerdere keren per dag
Scope	Betreffende team bedient de gehele organisatie

Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?

Voor onze organisatie geldt dat onze van product development is een hele zelfstandige en autonome club. En als je dan inzoomt op het echte dev gedeelte, dan is dat daarbinnen ook weer heel autonoom. Wij beheren en doen alles zelf. IT infrastructuur en sofware development en daar zitten hele capabele, technische **mensen die zelf de strategie bepalen om ook naar de toekomst toe de business te kunnen bedienen. Dus al dit soort keuzes, over hoe willen we werken en waarmee willen we werken, die liggen in het development team. Zij bepalen voor zichzelf wat er nodig is en wanneer. Daar ligt dus gewoon beslissingsbevoegdheid om dit te gaan doen en er komt geen groot besluitvormingsproces aan te pas, behalve dan binnen dat team zelf. De behoefte tot verbetering kwam voort uit wat fricties. Wij kwamen er gaandeweg achter dat de releases die we deden, inmiddels teruggebracht naar een keer per twee werken, toch nog steeds complex en groot waren. Daarnaast waren er wat problemen in de OTAP-straten, wanneer verschillende developers daarin werkten.** We zijn ook wat gegroeid in de laatste jaren, dus dan zit je elkaar ook vaker in de weg als je dat niet goed op orde hebt. **We deden weleens dinsdag een release en dan waren we de rest van de dinsdagmiddag bugs te fixen. Dat is frustrerend en kan anders**. Dus daar kwam het deels vandaan en **deels vanuit de wens van de business om snel waarde uitgeleverd te krijgen**. Als iemand van Sales iets bedenkt wat snel veel geld zou kunnen opbrengen, maar hij komt net op het verkeerde moment, dan kan het tot vier weken duren voordat dat wordt opgeleverd. **Met CDV wordt de werkwijze sneller en efficiënter en loop je minder risico. Wat mogelijk ook een reden was, is dat we met deze werkwijze gewoon met onze tijd en de ontwikkelingen in de wereld meegaan**. Net zoals dat we niet meer conform de waterval methodiek werken, maar Agile zijn gaan werken, waren we nu ook toe aan een volgende stap. **Bij de beslissing hebben we het veel gehad over het geschikt maken van de omgeving voor CDV.** Ik denk dat een van de grootste uitdagingen daarbij was, het op orde krijgen en goed gescheiden houden van de ontwikkelomgevingen.

Doelen/voordelen:

Het gaat erom in staat zijn snel waarde te leveren. Minder risico te lopen en daarmee efficiënter ons product development werk te doen. Je leert ook sneller, zowel op het businessvlak als op het technische vlak. Je levert iets op en je ziet al snel of dat werkt of niet. Je kunt dat dus ook weer snel aanpassen als het niet werkt.

Overzicht voordelen CDV:

Ik herken deze natuurlijk wel en denk dat de voordelen die ik genoemd heb hierin wel onder te brengen zijn. Ik denk dat er een relatie is tussen snel waarde opleveren en kortere time-to-market. Minder risico komt terug in nummer drie en nummer zeven. Efficiënter ontwikkelen zit in nummer 5. **Daarnaast zie ik ook wel wat relaties en overlap in deze voordelen. Nummer 2 en 6 hebben best wat met elkaar te maken.**

Dit overzicht had onze besluitvorming niet anders kunnen maken. developers zullen echter niet direct de kortere time-to-market als hun allergrootste winstpunt zien. Vanuit het

businessperspectief is dat natuurlijk wel zo. Als je dat aan developers uitlegt snappen ze dat ook wel, daar doen ze het uiteindelijk voor. **Bij ons was het dan niet nodig, maar dit overzicht van voordelen kan helpen bij organisaties waar mensen nog overtuigd moeten worden.**

Componenten:

De eerste herken ik sowieso natuurlijk, dit is waarom we het wilden doen. De tweede speelt natuurlijk een rol, wat ik al zei, je moet zorgen dat de omgevingen op orde zijn. **Dat is eigenlijk een randvoorwaarde om de stap te kunnen maken. Je softwarearchitectuur moet dat wel aankunnen.** Maar ik denk dat we daar al redelijk dichtbij zaten. Het aanpassen van de ontwikkelomgevingen was niet een hele grote klus. **Dit komt omdat we over het algemeen wel met onze tijd meegaan en kritisch naar onze architectuur kijken om vast te stellen wat nodig is. Nummer drie krijgt wel aandacht, we doen veel aan geautomatiseerd testen**. We hebben nog niet een QA engineer bijvoorbeeld, hier zijn nog wel verbeteringen mogelijk.

Zes is wel een interessante. Je moet dit wel uitleggen in het begin. Iedereen was gewend aan tweewekelijkse releases. Zowel de mensen intern als onze gebruikers. Voorheen gaven we uitleg op onze website aan onze gebruikers over de wijzigingen in de laatste release. We kregen daar dan ook direct feedback op. Nu we soms zelfs tien tot twintig releases op een dag doen, gaan we daar niet steeds dergelijke uitleg bij geven. **Dat is wel een onderwerp om over te communiceren.** Ook intern naar onze sales en marketing, wat we doen, waarom en wat dat voor ze betekent. Daarbij is overigens nooit twijfel geweest of we het zouden doen. Vanuit development hoeven we geen toestemming te vragen of iets dergelijks.

Of CDV voor nummer zeven nu echt iets heeft gebracht weet ik eigenlijk niet. Het heeft wel iets van waarde toegevoegd, maar we werkten al met het zogenaamde dual track roadmapping proces. Hierbij doen we discovery en delivery. Middels bijvoorbeeld A/B testing en usability testing stel je eerst vast wat de oplossing moet zijn voordat we het echt gaan bouwen. Dat is eigenlijk niet veranderd. Wanneer we nu een oplossing zien hebben we hem sneller gebouwd, maar deze werkwijze hadden we al.

In het kader van nummer acht, deden we eigenlijk al alleen maar rollforwards. Bij de tweewekelijkse releases deden we tussendoor wel hotfixes.

Organisatorische factoren gaven bij ons geen uitdaging. Wij zijn ook een relatief klein clubje. Ik kan mij voorstellen dat het voor grote organisaties met grote systemen wel een grote uitdaging is, waarbij je veel moet omgooien. Wij werkten al met multidisciplinaire teams die profiteerden van het feit dat we overstapten naar CDV. We zijn er ook langzaam naartoe gegroeid. Als je een transitie van a tot z moet doorlopen dan heb je op organisatorisch vlak wel erg veel te doen. Banken bijvoorbeeld zullen goed moeten kijken wat er organisatorisch mogelijk is. Bij ons was er een relatief gemakkelijk stap omdat de juiste mensen hier mandaat voor hebben.

Dit lijstje had wel kunnen helpen omdat het een mooi overzicht geeft van wat er nodig is. Bij ons is het niet zo noodzakelijk gebleken, maar anders had ik het graag willen gebruiken hoor.

Determinanten:

Ik waag te betwijfelen of een klant inderdaad niet altijd nieuwe software wenst. Ik denk dat de klant daar altijd bij gebaat is. Voor webapplicaties is het wel erg makkelijk en kan een klant zonder last over een nieuwe versie beschikken. Voor andere soorten software kan ik wel voorstellen dat het lastiger is. Maar ook dan moet er waarde zijn voor de klant.

Ten aanzien van het domein is het wel zo dat bepaalde onderdelen van ons landschap sterk verouderd zijn. Daarbij is het inderdaad wel lastiger om deze werkwijze van CDV op toe te passen.

Door technische beperkingen komt het zeker voor dat we voor onderdelen van ons landschap CDV nog niet toepassen. Bij architectuur zie je dat bijvoorbeeld onder aanpasbaarheid en testbaarheid terugkomen.
Op het gebied van cultuur hebben we wel een transitie achter ons liggen in de afgelopen jaren, wat geleid heeft tot werken met multidisciplinaire teams die het mandaat krijgen dat ze nodig hebben. Dit is wel een proces van jaren. Je moet er met z'n allen in geloven en het management moet het mogelijk maken. Dat is gebeurd. Ik zie het zeker als een hele belangrijke randvoorwaarde. Als je als ceo nog steeds wil bepalen wat er gebeurt, dan moet je dit niet doen.
Dat testen belangrijk is klopt hoor. Of de testinspanning zeer sterk toeneemt daar twijfel ik aan. Het kan daarvoor ook al erg belangrijk zijn in je werkwijze. Je geautomatiseerde teststraat moet wel op orde zijn.
Ik denk dat deze vijf het meeste wel bevatten. Ik kan niets bedenken wat ontbreekt. Dit laatste lijstje, net als de andere, helpen zeker om deze materie in het juiste kader te plaatsen.

Interview 5

Functie, periode	Programmamanager, traject van één jaar begeleid
Organisatie	Hypothecaire dienstverlening, business to business, ca. 800 medewerker in dienst destijds
Status CDV	Deelnemer kwam bij de organisatie werken toen besloten was tot implementatie en heeft het eerste jaar van de implementatie meegemaakt. Ca. 7-8 jaar geleden.
Softwareontwikkelproces	Volledig in eigen huis, met deels ingehuurde developers.
Releasefrequentie	1-2 keer in de maand, dat verschilde per klant.
Scope	Team van 40 developers kreeg als eerste CDV geïmplementeerd.

Status CDV:
Wat de organisatie eigenlijk deed was dienstverlening voor meerdere partijen. Daarmee werd eigenlijk een soort white label product gecreëerd. In de kern was er bepaalde software, maar vaak werd dat wel met maatwerk aangevuld voor verschillende klanten. Hierdoor was er een enorm uitgebreid applicatielandschap. De releasefrequentie verschilde per klant. Het kon ook goed zo zijn dat een nieuwe versie van de software door de ene klant al werd gebruikt, maar door een andere nog niet. De frequentie was ook laag omdat het moeizaam was om alles kwalitatief op te leveren. Voor bepaalde klanten ook wel een keer per half jaar.

Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?
De organisatie was een operationele dienstverlener, waarbij je de verwachting zou hebben dat ze erg gebaat zijn bij CDV. **Toch is het begonnen als een IT feestje. Vanuit IT zijn de eerste stappen gezet. Destijds was het nog een erg nieuwe ontwikkeling en ging het gepaard met de nodige onzekerheid en risico. Toch hadden de CEO en CIO een visie en vonden ze dat we dit moesten doen.** De reden dat ze dit willen is denk ik tweeledig. **Enerzijds is het natuurlijk goed voor de organisatie, maar wat ik wel zie is dat het misschien op persoonlijk vlak goed is voor deze mensen. Een interessante ontwikkeling die goed staat op hun cv. Ik denk dat je dat vaker ziet in organisaties**. **Nu zou je het overigens om andere redenen doen. Om bij te blijven bij je concurrenten. Destijds werd CDV ook als een middel gezien om de marktpositie te vergroten. De overtuiging was dat dat niet ging lukken met de huidige systemen. Daar zaten teveel fouten in en klanten waren niet voldoende tevreden. Althans, de fouten zaten in de softwareopleveringen en dat droeg bij aan klantontevredenheid. Daardoor ging er veel aandacht uit naar het herstellen en onderhouden van de relatie met de klant. Die aandacht kon niet naar het werven van nieuwe klanten. Daar zit eigenlijk de relatie tussen het marktaandeel vergroten en CDV.** Maar men wilde wel de grootste worden en destijds betekende dat dat een bepaalde concurrent verslagen moest worden. Wellicht was dat deels ook wel een politieke of persoonlijke agenda. **Er was ook wel degelijk een operationeel IT probleem en dat wilde men ook verbeteren. Los van de ambitie om te groeien, was het in stand houden van de bestaande dienstverlening al uitdagend. Daarnaast was er ook een personele overweging. Er waren veel verschillen in de software ten behoeve van de vele verschillende klanten. Kennis hiervoor zat doorgaans in de hoofden van mensen. Als je dat gaat automatiseren, dan haal je die kennis uit de hoofden van mensen en komt het bijvoorbeeld in een script terecht.**

Doelen/voordelen, overzicht voordelen:
Ik herken deze voordelen allemaal. Ik denk dat we ze goed kunnen ranken en dat ik er net ook wel de meeste heb benoemd. Wat hoog voorop stond was kortere time-to-market, volgens mij noemde

ik dat ook. Dat bedoelde ik met het feit dat releases lang duurden en ook foutgevoelig waren. De klant wilde vaak wel eerder nieuwe versies hebben. De derde hangt daar natuurlijk sterk mee samen. **De eerste en de derde zijn het belangrijkst. De vierde zie ik ook, maar meer als gevolg daarvan**. Ik denk dat dat de grootste drijfveren zijn geweest. Voor vijf is misschien ook wel wat te zeggen, maar dat gaat eigenlijk vanzelf en zie ik meer als een gevolg. We kwamen vanuit een probleemsituatie, waarbij we niet veel aan vernieuwing toe kwamen. Door het model van whitelabeling wilde iedere klant het net weer even anders, waardoor er heel veel verzoeken waren waar men niet aan toe kwam. De capaciteit die daarvoor overbleef was erg laag, omdat teveel aandacht moest naar het herstellen van fouten. Door dat op te lossen krijg je meer capaciteit voor doorontwikkeling. **Dus ja, dan kun je ook stellen dat de ontwikkelproductiviteit omhoog gaat en logischerwijs kun je dan ook sneller innoveren. Voor onze organisatie gold dat al deze voordelen relevant waren, maar ik zie dat vooral terugkomen in de onderlinge relaties.** De drie die ik noemde zijn wel de primaire.
Bij de adoptiebeslissing is ook wel afgewogen wat de investering kost en wat het zou opleveren. Die oplevering zat in de vraag welke pijnen verminderd zouden worden. En juist daarom vind ik dat verhoogde klanttevredenheid erg zwaar meetelt. We wilden namelijk geen klanten verliezen. Nummer drie heeft daar erg veel invloed op. Deze voordelen werden wel gebruikt bij het opstellen van een business case, ook nummer 1 en nummer 6, maar ik weet niet meer zo goed hoe dat precies is gegaan. **Kostenbesparing op personeel was daar ook een overweging bij, die komt wellicht terug in nummer 5. Bij de onderbouwing van de business case was de personele kostenbesparing wel een belangrijk onderdeel.**

<u>Componenten:</u>
Snelle en frequente release, zeker. Flexibel productontwerp en architectuur, ik denk dat in eerste instantie niet. **Ik denk dat architectuur niet zo'n belangrijke rol of invloed had in die periode. Dat is nu anders. Het is wel waar wat hier staat, maar ik denk dat dat toen niet zo tussen de oren zat.** Het werd misschien wel als een factor gezien, maar als ik zie hoe dat nu gaat is dat veel prominenter geworden. **Architectuur speelt namelijk wel degelijk een rol en organisaties gaan daar tegenwoordig veel bewuster meer om. Nummer drie, vier en vijf komen wat mij betreft een beetje bij elkaar. Dit heeft wel gespeeld. Ik zie hierin het automatiseren van de delivery pipeline, dus van provisioning tot en met deployment. Waarmee geautomatiseerd testen toeziet op kwaliteit.** Dit zijn een beetje de pijlers van wat we gedaan hebben. De betrokken klant sloeg niet zozeer op het ontwikkelproces, maar de klant kreeg wel eerder geleverd wat werd gevraagd. Vroeger feedback ontvangen is niet een van de pijlers geweest om deze beslissing te nemen. **De klant speelde wel een belangrijke rol bij de beslissing.** Maar bij feedback vind ik dat je de klant er veel dichter op moet laten zitten en dat was gewoon niet zo. **De verhoogde klanttevredenheid moest vooral voortkomen uit de verbeterde kwaliteit en betrouwbaarheid van de release. Niet zozeer door een hogere releasefrequentie.** Het speelde wel een rol, maar we wilden vooral vaker opleveren om de lange lijst van wijzigingsverzoeken weg te werken. Met software die foutloos eens in de drie maanden, in plaats van eens in de zes maanden met fouten, werd vernieuwd waren de meeste klanten eigenlijk al tevreden. Het ging vooral om het voorkomen van de fouten en er was geen behoefte aan een hogere frequentie. **Sterker nog, een klant moet ook een acceptatieproces in dus het kan ook een nadeel zijn. De software die destijds werd geleverd, was voornamelijk on-premises, dus bij de klant geïnstalleerd.** Daarnaast was er ook sprake van dat het volledige proces was uitbesteed aan onze organisatie. Vandaar dat het onderdeel Operations zo groot was.
Ik denk dat de kennis in dit overzicht wel voor een andere situatie had kunnen zorgen. Destijds was het gedachtegoed nog vrij jong en ik denk dat veel onderdelen nog helemaal niet zo scherp in beeld waren. De organisatorisch factoren vallen daarbij het meest op. Het was te lang een IT

feestje. De afdeling Operations zou er veel van merken en dat was het leeuwendeel van het bedrijf. 600 van de 800 medewerkers destijds. **Men was vergeten die medewerkers hierin mee te nemen, dus dat moest hersteld worden.**

Determinanten:

Ja herkenbaar, de klant wil niet altijd een hogere releasefrequentie. Wat we eigenlijk niet wilden is dat de klant merkte dat we met deze verandering bezig waren, anders dan dat de kwaliteit van de software omhoog ging. Daar hoorde dan ook wel bij dat de releasefrequentie iets omhoog ging. **We zijn begonnen in de domeinen midoffice en backoffice. Hier was men destijds wel bewust van, maar wist niet voldoende welke impact daaraan verbonden was.** De start met midoffice software kwamen tools beschikbaar die zorgden voor goede resultaten die erg stimuleerden om door te gaan. **Toen we bij het backoffice domein uitkwamen, met veel meer batchverwerkingsprocessen bleek het wel ingewikkeld en ontstond de vraag hoe dit te doen**. Gaandeweg dat traject is steeds onderzocht wat de oplossing moet zijn. **En ook al weet je dat je dergelijke problematiek gaat raken, denk ik dat er weinig bewustzijn op is geweest.** En eigenlijk is dat gek. **Want het domein heeft, samenhangend met de architectuur daar heel veel invloed op. De architectuur heeft erg vaak in de weg gezeten bij het implementeren van CDV. Vooral in het backoffice domein, door de batchverwerkingsprocessen.** Voor die software is testautomatisering lastig. **Het is wel voorgekomen dat we destijds voor bepaalde applicaties CDV nog maar even niet hebben toegepast.**

Ik denk dat cultuur een van de belangrijkste factoren is om rekening mee te houden. Je hebt namelijk te maken met een bepaalde cultuur in een organisatie. Je moet je daarbij afvragen hoe bereidwillig iedereen is, medewerkers maar vooral ook het management, om deze verandering in te zetten. Dit mag je niet onderschatten, want een bestaande cultuur kan erg hardnekkig zijn. En cultuur verander je op zich niet. Je kunt gedragingen veranderen en als resultaat daarvan zou ze op zeker moment kunnen vaststellen dat de cultuur wat veranderd is. **Verschillen binnen een organisatie zijn hierbij wel waarneembaar. In het ene team of subteam kan een andere cultuur bestaan dan in het andere. Dat gaat ook organisch, dezelfde soorten mensen zoeken elkaar toch een beetje op. Een groep mensen kan erg openstaan voor verandering en innovatie, terwijl een andere groep erg behoudend is**. Dit komt ook terug in rollen. **Een ontwikkelaar zal eerder willen innoveren dan een beheerder. Er is ook wel rekening gehouden met de factor cultuur bij het nemen van de adoptiebeslissing. Daarbij is heel bewust gekozen om het tegelijk te introduceren met Agile werken.**

Testen heeft absoluut invloed gehad op de adoptiebeslissing. Waar het opleveren van een server eerst bijvoorbeeld een week duurde, met fouten. Was het na het traject met een druk op de knop na vijftien minuten voor elkaar, foutloos. De tests daarbij waren geautomatiseerd. Het maken van de geautomatiseerde tests zelf, daar is eigenlijk in het begin niet goed naar gekeken. Er is niet genoeg rekening gehouden met hoeveel werk dat is. **Het is vooral een factor omdat je niet moet onderschatten hoeveel werk testautomatisering is. Bij een nieuw systeem is dat niet zo'n probleem want dan groeit het mee. Voor een bestaand systeem moet je eerst een hele berg verzetten, je hebt echt een enorme achterstand. Vanwege de factor testen hebben we wel besloten om voor applicaties nog even niet de CDV werkwijze te hanteren. Dit is eigenlijk hetzelfde als ik hiervoor gezegd hebt over de backoffice applicaties. Vanwege de architectuur dus. Ik denk dat deze determinanten een rol hebben gespeeld bij de beslissing. Wat meer bewustzijn had zeker kunnen helpen. Het is een fijn lijstje dat je kunt hanteren**. Dus in plaats van dat je een zoektocht krijgt naar wat je moet doen, kun je dit als handvat gebruiken. Destijds was dit allemaal nieuw en was externe hulp nodig. Het was nog heel lastig om er een te vinden. **Ik zie niet een determinant die mist in dit overzicht.**

Interview 6

Functie, periode	Manager development, afgelopen twaalf jaar
Organisatie	Sofwareleverancier, ca. 50 developers in dienst voor betreffende productlijn
Status CDV	CI geïmplementeerd. CDV in principe ook, maar niet tot en met het gebruik door de klant. Sinds eind 2017 Agile/CI/CD ambitie
Softwareontwikkelproces	Software ontwikkelen is de core business
Releasefrequentie	Iedere zes weken voor de klant, maar intern iedere twee weken
Scope	Deel van de organisatie met focus op een productlijn

Status CDV:

In 2017 zijn we ons gaan oriënteren op Agile werken. Tot die tijd werkten we eigenlijk nog conform watervalmethode. Eind 2017 hebben we een pilot gedaan en in 2018 hebben we besloten het voor alle development teams toe te passen. **Daarmee creëerden we een snelheid waarvoor een verregaande automatisering nodig is. Je loopt bijvoorbeeld aan tegen de grenzen van je testcapaciteit.** Dus om dat ritme van sprints van twee weken vast te houden, moet je heel veel dingen automatiseren die je tot dat moment nog niet geautomatiseerd had. **Dus toen zijn we met CI gestart als facilitator van Agile werken en om tegemoet te komen aan de flexibiliteit die klanten van ons vroegen. Om van Agile werken een succes te maken moesten we CI gaan doen.** Na de keuze om Agile te werken zijn we ons meer gaan verdiepen in CI. **Het is niet zo dat we nog niets deden, we hadden delen al wel geautomatiseerd,** bijvoorbeeld een deel van de regressietest en het installeren van programmatuur van de ene omgeving naar de anderen. **Later hebben we in bredere context gekeken naar CI en hebben we het fenomeen volledig toegepast,** zodat we elke dag een oplevering in een testomgeving zouden kunnen doen. Daar zit ook iets bijzonders in overigens. Wij hebben wel een omgeving die we productieomgeving noemen, maar feitelijk is dat ook een testomgeving. We beschikken over heel veel verschillende testomgevingen, voor verschillende doeleinden en we gebruiken er een om de live situatie van een klant zoveel mogelijk na te bootsen. Waarin we bijvoorbeeld batches draaien en transacties doorvoeren. **De stap naar CI kwam aanvankelijk misschien enigszins onbewust. Het zoveel mogelijk genereren van software, zoveel mogelijk controleren van standaarden en richtlijnen, geautomatiseerde tests, het geautomatiseerde proces om omgevingen op te tuigen, dat zijn dingen die je in een waterval ook prima kunt gebruiken. Elke professionele organisatie is met dat soort dingen bezig. Toen wij naar een ritme van twee weken wilden, zijn we ons gaan verdiepen in de principes die daarover worden afgesproken in de wereld om ons heen. Dat blijkt dan CI te heten.** Maar toen we er concreet mee aan de slag gingen zaten daar nog wel wat nieuwe dingen bij.

In 2017 hebben we geformuleerd dat we op een andere manier willen gaan werken en welke dingen we wilden veranderen. **Daarbij wilden we meer met klanten samenwerken, we wilden duidelijker prioriteren en ook samen met klanten prioriteren**. **We wilden ook de kwaliteit verhogen** en **flexibeler zijn, wat betekende tussentijds prioriteiten wijzigen**. Agile werken was daar de oplossing voor. We zijn begonnen met een bepaald team. Dat kreeg steeds meer aandacht, waardoor we ook onderzocht hebben of niet de hele organisatie op die wijze kon gaan werken. **Die beslissing hebben we uiteindelijk samen met de medewerkers gemaakt. Je moet daarover wel in gesprek gaan. Je kunt dat niet zomaar afdwingen als management.**

Voorheen werkten we met twee major release per jaar, aangevuld met patches. Nieuwe functionaliteit zat alleen in de major releases. Toen we kortcyclischer gingen werken vervaagde dat een beetje en kwam er ook functionaliteit in de patches terecht. Vanaf de Agile transitie werken we in sprints van twee weken, waarbij we iedere zes weken een release aan de klant opleveren. **We zijn ook wel in staat om software iedere twee weken aan de klant te releasen, maar klanten willen dat**

over het algemeen niet. Die hebben ook acceptatiewerk aan hun kant en hebben al een flinke uitdaging aan de huidige frequentie, ook gezien waar we vandaan komen.

Kunt u iets vertellen over de manier waarop de organisatie de beslissing genomen heeft om de werkwijze van CDV te implementeren? Hoe is dat gegaan?
We houden twee keer per jaar een klanttevredenheidsonderzoek. De feedback die daaruit kwam beschreven een grote, logge organisatie. Wijzigingsverzoeken duurden erg lang, afstemming tussen klanten. Daarna nog bouwen, testen, enz. Dit lag niet alleen aan ons, maar ook aan de klanten vind ik. Het begin en einde van proces konden namelijk ook beter en daarin zit de communicatie met de klant. **In het midden van het proces echter, bleek toch ook wel dat wij niet opleverden wat gevraagd werd en is het aan ons. We zijn toen gaan kijken hoe we het development proces konden versnellen.** We wilden meer met klanten samenwerken, de interactie vergroten en meer snelheid erin brengen. Concluderend zagen we drie behoeften. **We wilden anders met klanten omgaan**, **intern sneller werken**. **Daarnaast zagen we de wereld om ons heen steeds sneller veranderen en wilden we flexibeler zijn om daar tegen bestand te zijn. Daarom is besloten om over te gaan tot Agile werken. CI is later het middel gebleken om dat voor elkaar te krijgen en die snelheid te bereiken.** We gingen intern scrummen, stelden een governance model voor de omgang met klanten. Stuurgroepen en werkgroepen met klanten werden ingericht. We doen het nu veel meer samen. Deze keuzes worden allemaal niet op hetzelfde moment gemaakt. Eerst hebben we besloten Agile te gaan werken. **Vervolgens hebben we besloten om volledig tot CI over te gaan. Als je CDV ziet als het grotendeels geautomatiseerd samenstellen op opleveren van een release, dan denk ik dat we dat voor 95% hebben bereikt. Als je daarin meeneemt dat het ook daadwerkelijk op een productieomgeving terecht moet komen, dan zijn we daarin afhankelijk van onze klanten.** Dus die stap gaan we niet bereiken denk ik op korte termijn**. De vraag is of je dit voor dergelijk mission critical systeem moet willen. En dat is een overweging die vooral aan de klant is.** Ik zie het niet snel gebeuren dat we grote wijzigingen automatisch pushen naar een productieomgeving. **Wat lastig is, is dat klanten niet willen dat wij toegang hebben tot hun data. Klanten zouden wel kunnen leren van onze ervaring met betrekking tot automatisch testen en deployen**. Wij hebben ook klanten die onze oplossing vanuit de cloud afnemen. De stap om daar een geautomatiseerde installatie uit te voeren, vergelijkbaar met hoe wij dat doen voor onze testomgevingen en interne productieomgeving, is niet zo heel groot. De klant kan dan wel met hun eigen data testen. Daarover zijn we wel in gesprek. **Als ik droom over de toekomst zie ik wel voor me dat klanten nieuwe versies naar hun acceptatieomgeving gepusht krijgen en dat wij een sein terugkrijgen als de acceptatietest succesvol is verlopen. Vervolgens zouden we dan automatisch kunnen releasen naar productie.**

Doelen/voordelen:
Vroegtijdige feedback van klanten, waardoor we de juist dingen bouwen. De mogelijkheid om flexibel te kunnen ingrijpen bij veranderende prioriteiten. Snelheid. Ik denk dat dat de belangrijkste zijn.

Overzicht voordelen CDV:
Ja, ik herken ze allemaal wel. Deze argumenten hebben we ook wel gebruikt om dit aan de man te krijgen.

Componenten:
De eerste is een van de aanleidingen geweest om Agile te gaan werken en om CI in te voeren. **Voor wat betreft de tweede, ons softwareproduct bestaat al een tijdje, maar het is niet zo dat we onze**

complete softwarearchitectuur overhoop hebben moeten halen om dit te realiseren. De architectuur van onze software bleek eigenlijk prima in staat om de concepten van CI te implementeren. Dat wil niet zeggen dat het optimaal is. Ik denk dat er flexibelere producten bestaan, waarbij je bepaalde stukken wel en andere niet kunt deployen. **Onze keuzevrijheid hierin is beperkt, omdat dingen met elkaar kunnen samenhangen. Ik weet niet of dit komt door verouderde architectuur, het heeft ook met de complexiteit van het systeem te maken.** Onze software is in hoge mate configureerbaar. Klanten hoeven daardoor niet alle functionaliteiten in gebruik te nemen. Maar ook functionaliteiten die niet gebruikt worden kunnen tot regressie leiden. Daarom testen we erg zorgvuldig. **Met een systeem dat zo complex is als het onze, los van de architectuur is flexibiliteit moeilijk bereikbaar. Als je iets bouwt dat volledig op zichzelf staat is er niet zoveel aan de hand, maar wanneer je te maken hebt met kritieke verwerkingen dan moet je steeds controleren of alles nog goed werkt.** Deze complexiteit van het systeem komt eigenlijk door de omvang en de mate waarin functionaliteiten met elkaar samenhangen.
De derde, vierde en vijfde zie ik wel, maar dat zijn ook onderwerpen die we al deden. De betrokken klant was een belangrijke aanleiding. Nummer 8 speelde niet echt een rol. Nummer negen overduidelijk wel. **Nummer tien, deze transitie heeft wel gezorgd voor commitment, eigenaarschap en enthousiasme bij medewerkers. We hadden daar ook erg op gehoopt. We hebben deze verandering zorgvuldig moeten aanpakken. We zijn namelijk een development club. We kunnen het ons niet permitteren om bij dergelijke verandering bijvoorbeeld de helft van onze mensen te verliezen.** Dan hadden we onze voorgenomen doelen niet bereikt. In de basis heeft het management ervoor gekozen dit te willen gaan doen en vervolgens een aanpak gekozen waarbij de mensen optimaal enthousiast mee zouden gaan werken.
Samenvattend: 1, 6, 9 en 10 zijn de belangrijkste geweest.

<u>Determinanten:</u>
De klant is inderdaad een erg belangrijke determinant, vooral in positieve zin. Ze waarderen de versnellingen die we hebben gemaakt, maar de laatste stap is niet altijd gewenst. Klanten willen toch nog zelf testen en ook vooral hun eigen inrichting testen. Voor de laatste stap naar CDV waarbij je ook automatisch kunt deployen naar de productieomgeving is de klant op dit moment de beperkende factor.
In ons geval is sprake van een back office applicatie, ondersteunend aan het primaire proces van de klant. Om een werkende productieomgeving in te richten, moet daar een database voor zijn, er moet een applicatieserver zijn, onze software moet er zijn, omringende systemen moeten daarmee kunnen interfacen. Op het moment dat wij wijzigingen doorvoeren, kan dat daarop impact hebben. **Dus als je dat allemaal tot het domein rekent, dan is dat zeker medebepalend voor de drempel die is er is om echt aan CDV te doen.** Maar bij ons intern is dit gelukt en kunnen we dit wel allemaal automatisch installeren. Het kan dus wel.
Onze applicatie heeft inderdaad wel het karakter van een monoliet. Het zijn grote stukken samenhangende programmatuur. Ons product is in relatief hoge mate configureerbaar. Hoe meer configuratieopties je biedt, hoe complexer het wordt om dit te realiseren en te onderhouden. En hoe meer combinaties van configuraties er door klanten gebruikt kunnen worden. Hierdoor wordt het steeds complexer om te weten wat er bij een klant staat, hoe het gebruikt wordt en hoe je issues moet oplossen. Toch heeft dat bij ons allemaal niet in de weg gezeten. **Ondanks deze eigenschappen waren we uitstekend in staat te komen tot waar we nu zijn en zit de architectuur niet in de weg bij een CDV implementatie. In het algemeen denk ik het wel dat architectuur een determinant is.**
Ja, ik denk wel dat cultuur een determinant is. Vanuit mijn rol als manager ben ik erg doordrongen van wat hier staat. Het moeten zelforganiserende teams zijn. Niet zelfsturend, die discussie hebben

we weleens gehad, maar dat zou betekenen dat ze zelf hun doelen kunnen bepalen. Het management bepaalt doelen en kaders en het team kan zelf het werk organiseren om die doelen te behalen. Die ruimte willen we ze ook heel sterk geven. **Ik probeer invulling te geven aan dienend leiderschap en vooral faciliterend te zijn**. Ik zie ook wel dat dat mensen motiveert. **Als management moet je dit wel aandurven. Als het hoger management niet accepteert dat je die teams het vertrouwen en de zelfstandigheid geeft, dan is het eigenlijk kansloos.** Die omslag hebben wij ook wel gemaakt. Wat je in een watervalaanpak nog wel ziet en bij Agile niet, is dat je puur analisten, bouwers en testers hebt. Wij willen wel graag dat iedereen deze skills kan leveren. Dat zie je wel verkeerd gaan bij andere organisaties. Bijvoorbeeld analisten die per se alleen analist willen blijven. Daar hebben wij er ook wel een paar van gehad en die zijn afgevallen. **Nu is iedereen bij ons developer en moet in ieder geval ook kunnen programmeren. Niet iedereen hoeft de grootste development expert te zijn, maar voor iemand die alleen maar kan analyseren of testen is geen plek meer.**
Wij streven naar 100% geautomatiseerd testen, wat betekent dat hetgeen dat getest wordt volledig automatisch moet gebeuren. Een 100% test is niet realistisch, het blijft een risicoafweging. **Verder ben ik het eens met wat hier staat. Je moet vooraf zeker bewust zijn van het feit dat je geautomatiseerd testen nodig hebt, anders ga je dit niet realiseren.**
Ik herken alle vijf deze determinanten. Ik denk dat je over alle vijf heel goed moet nadenken of je deze stap kunt zetten. Klant, domein, cultuur en testen dat zijn dingen waar wij heel bewust van hebben gezegd dat is geen enkel issue, we gaan dit gewoon doen. Architectuur hebben we niet zo als een probleem gezien. We hebben er wel over nagedacht, maar het speelde minder een rol. **We hebben deze determinanten niet zo expliciet onderkend en stuk voor stuk beoordeeld, maar nu ik ze zo zie staan denk ik wel dat het goed zou zijn geweest om er op die manier naar te kijken.**
De reguliere managementrollen verdwijnen. Dat is een belangrijke organisatorische factor. Je zou dat kunnen zien als onderdeel van cultuur, maar misschien staat het wel op zichzelf. Het organogram wijzigt gewoon en daar moet je wel toe bereid zijn.

Bijlage 6: Totaaloverzicht coderingen

Nr.	Code 1	Code 2	Tekst	Oorsprong (interview, deel)
1	Domein	Neutraal	Wij hebben meerdere softwareontwikkelteams, vaak extern ingehuurd, die ieder verantwoordelijk zijn voor een bepaald domein of verschillende services.	1, status CDV
2	Cultuur	Steun	Het staat de teams min of meer vrij om hun eigen werkwijze toe te passen.	1, status CDV
3	Doelen	Bekend	Het doel van de organisatie is om zoveel mogelijk waarde uit de kosten voor deze teams te halen.	1, status CDV
4	Nieuw inzicht	Agile enabler	CDV is een middel om meer Agile te werken	1, status CDV
5	Nieuw inzicht	Algemene ontwikkeling	Het idee is dus wel om CDV overal mogelijk te maken. Maar soms zijn we gewoon nog niet zover. En dat komt omdat mensen de kennis of de vaardigheden nog niet hebben, of omdat de omgeving waarin zij werken, bijvoorbeeld SAP, dit nog niet mogelijk maakt.	1, status CDV
6	Nieuw inzicht	Onbewuste keuze	Je zit fout door te denken dat een organisatie dit bewust introduceert.	1, adoptiebeslissing
7	Nieuw inzicht	Algemene ontwikkeling	We zijn bezig om een omgeving te creëren waarin we efficiënt kunnen werken.	1, adoptiebeslissing
8	Nieuw inzicht	Onbewuste keuze	We denken niet actief na over organisatorische kansen en problemen waarbij CDV de oplossing is.	1, adoptiebeslissing
9	Cultuur	Steun	Dat doen zij van zichzelf uit en dat kunnen ook partijen uit de markt zijn.	1, adoptiebeslissing
10	Domein	Steun	Uiteraard zoeken we dan naar een team dat op een moderne manier software ontwikkelt. Maar er zijn meerdere criteria, zoals bepaalde domeinkennis. Een specialist in mobile apps kun je niet gebruiken voor replenishment. Soms moet je bij je keuze compromissen maken. We moeten soms contracten aangaan met teams die niet op een absoluut moderne manier werken, maar die wel de absolute expert zijn in hun domein.	1, adoptiebeslissing
11	Nieuw inzicht	Domein bepaalt cultuur	Retail kent deels echt nog een erg traditionele manier van softwareontwikkeling. Sommige onderdelen zoals het hele kassa-applicatielandschap,	1, adoptiebeslissing

			datawarehouselandschap, de hele replenishment, alles wat met logistiek heeft te doen, dat is zo ouderwets dat het niet mogelijk is om CDV toe te passen	
12	Cultuur	Steun	En de mensen spelen hierbij ook deels een rol. Die hebben ergens in de jaren ’90 de software gebouwd, dus daarmee hebben ze een oude manier van denken.	1, adoptiebeslissing
13	Architectuur	Steun	Daarnaast denk ik dat een architectuur geschikt moet zijn om CDV toe te passen. En veel van deze applicaties volgen deze basisprincipes niet, dus die zijn technisch niet in staat het te doen. Als je een grote monoliet hebt, is het vaak niet mogelijk geautomatiseerd te testen.	1, adoptiebeslissing
14	Doelen	Bekend	Op businessniveau is efficiënter werken het doel	1, doelen/voordelen
15	Doelen	Afwijkend	Je wil een uitdagende werkomgeving voor nieuwe en jonge medewerkers. Omdat wij denken met deze mensen een meer flexibele en Agile manier van werken gedaan te krijgen en onszelf te blijven uitdagen.	1, doelen/voordelen
16	Doelen	Bekend	sneller kunnen reageren wanneer er iets misgaat	1, doelen/voordelen
17	Doelen	Bekend	complexere releases vermijden	1, doelen/voordelen
18	Relatie	Voordelen	Maar dat zie ik echt als tactische doelen, onder het grotere geheel.	1, doelen/voordelen
19	Rol voordelen		De opsomming van de voordelen zou op zich zeker kunnen helpen. Echter, ik zie de uitdaging in het introduceren van CDV niet zozeer in het overtuigen van mensen dat het zinvol is.	1, overzicht voordelen
19	Nut overzicht		De opsomming van de voordelen zou op zich zeker kunnen helpen. Echter, ik zie de uitdaging in het introduceren van CDV niet zozeer in het overtuigen van mensen dat het zinvol is.	1, overzicht voordelen
20	Nieuw inzicht	verschil klant/leverancier	Wij zijn dus meer bezig met het integreren van standaardsoftware dan het bouwen van software.	1, overzicht voordelen
21	Architectuur	Steun	En in de onderdelen waar wij geen technische architectuur beperkingen hebben, daar doen wij het al.	1, overzicht voordelen
22	Rol componenten		Ik denk eigenlijk dat je dit allemaal nodig hebt.	1, componenten

23	Nut overzicht		Dit overzicht laat eigenlijk heel goed zien waar wij mee zitten. Wij zitten met noodzakelijkheden en beperkingen buiten onze context waardoor sommige van deze componenten ons niet gegeven zijn en wij CDV dus niet kunnen inzetten.	1, componenten
24	Domein	Steun	Je kunt hier niet alle domeinen hetzelfde in behandelen.	1, determinanten
25	Domein	Steun	Ons replenishment domein is er gewoon niet geschikt voor.	1, determinanten
26	Domein	Neutraal	Wij hebben waarschijnlijk een stuk 20 domeinen, maar helaas hebben wij geen adequaat overzicht van al onze domeinen.	1, determinanten
27	Domein	Steun	Grofweg zie je dat de front-end applicaties, denk bijvoorbeeld aan webapplicaties met een focus op business to consumer, erg modern zijn. Daarnaast zie je dat voor alle back-end applicaties zoals SAP of een datawarehouse, onze klant dingen eigenlijk helemaal niet snel wil veranderen.	1, determinanten
27	Klant	Steun	Grofweg zie je dat de front-end applicaties, denk bijvoorbeeld aan webapplicaties met een focus op business to consumer, erg modern zijn. Daarnaast zie je dat voor alle back-end applicaties zoals SAP of een datawarehouse, onze klant dingen eigenlijk helemaal niet snel wil veranderen.	1, determinanten
27	Relatie	determinanten	Grofweg zie je dat de front-end applicaties, denk bijvoorbeeld aan webapplicaties met een focus op business to consumer, erg modern zijn. Daarnaast zie je dat voor alle back-end applicaties zoals SAP of een datawarehouse, onze klant dingen eigenlijk helemaal niet snel wil veranderen.	1, determinanten
28	Nieuw inzicht	Domein bepaalt cultuur	Dat is niet alleen bij de klant, maar ook bij de leverancier zo.	1, determinanten
29	Cultuur	Steun	Cultuur speelt hierbij absoluut een rol. Je ziet daar vaak culturen van mensen die het al 20 jaar doen en die niet zo houden van verandering. Omdat ze in het verleden al meegemaakt hebben dat veranderingen slecht voor hen waren.	1, determinanten

30	Architectuur	Steun	De onderwerpen die staan benoemd bij architectuur, die zijn grote monolieten doorgaans niet goed gegeven.	1, determinanten
31	Domein	Steun	Het front-end domein is over het algemeen gewoon nieuwer.	1, determinanten
32	Domein	Steun	Bij domeinen die nog niet zo lang bestaan zie je dat ze moderner ingericht zijn.	1, determinanten
33	Nut overzicht		Voor alle determinanten geldt dat ze weleens de stap naar CDV in de weg zitten.	1, determinanten
34	Relatie	determinanten	Architectuur en domein zijn sterk aan elkaar gekoppeld en zijn voor ons de belangrijkste determinanten.	1, determinanten
35	Nut overzicht		Deze determinanten helpen mij te beschrijven waar we staan, ik mis er geen.	1, determinanten
36	Nieuw inzicht	Domein bepaalt cultuur	Dat komt aan de ene kant door de volwassenheid van de relatie met de leverancier en aan de andere kant de tooling die zich er niet zo goed voor leent om snel zoiets op te starten.	2, status CDV
37	Doelen	Bekend	een week lang doortesten was voor elke release. En dat heel veel mensen dat moesten doen, waar het echt veel nachtwerk ook werd	2, adoptiebeslissing
38	Doelen	Bekend	Er gebeurde zoveel online en ook zoveel klantvragen daarover waar we eigenlijk enerzijds niet in konden voldoen qua snelheid destijds	2, adoptiebeslissing
39	Doelen	Bekend	Als we iets deden dan kostte dat heel veel geld en heel veel tijd.	2, adoptiebeslissing
40	Cultuur	Steun	een andere mindset te geven aan heel veel mensen	2, adoptiebeslissing
41	Cultuur	Steun	externe factoren. In die zin, ik was nieuw, mijn manager was nieuw en er is een andere web agency aangehaakt, dus er kwam heel veel vers bloed	2, adoptiebeslissing
42	Nieuw inzicht	kosten	conversieverhogend zouden werken en kosten zouden besparen. Dat neigt misschien naar een business case. Ik weet niet of er nog een hele zware business case lag voordat ik en mijn manager binnenkwamen.	2, adoptiebeslissing
43	Doelen	Bekend	ja dingen sneller live te zetten	2, adoptiebeslissing
44	Doelen	Bekend	dan de kwaliteit ook niet altijd ten goede kwam	2, adoptiebeslissing
45	Nieuw inzicht	kosten	vrij prijzig geintje was voor datgene wat er in een release zit. Ten opzichte van de uitgave, wordt er weinig waarde opgeleverd.	2, adoptiebeslissing

46	Rol voordelen		In theorie zou ik zeggen alles een rol speelde, maar de praktijk is vaak weerbarstiger.	2, doelen/voordelen
47	Rol componenten		Nummer 1 uiteraard. Nummer 3. Nummer 9 is ook wel belangrijk geweest toentertijd. Nummer 8 was ook nog wel relevant	2, componenten
48	Relatie	Voordelen	Verhoogde klanttevredenheid komt voort uit in staat zijn eerder iets te leveren wat de klant wil, of eerder iets te fixen waar de klant last van heeft.	2, componenten
49	Nieuw inzicht	Onbewuste keuze	We waren ons ten tijde van de beslissing niet bewust van al deze voordelen. Je kiest een paar hoofdargumenten uit om mensen mee te krijgen om dit te gaan doen.	2, componenten
50	Nieuw inzicht	Overtuigen	We zijn niet begonnen met tien voordelen of iets dergelijks. Dat verkoopt zo slecht. Ik denk dat je een aantal voordelen moet uitkiezen die het meest tot de verbeelding spreken voor de organisatie.	2, componenten
51	Nieuw inzicht	Overtuigen	Je moet argumenten uitkiezen die snel iets opleveren, dan kun je mensen meekrijgen.	2, componenten
52	Nut overzicht		Ja, ik denk dat dit overzicht zou kunnen helpen bij het maken van de adoptiebeslissing. Men brengt te gemakkelijk buzzwords naar voren. Even een stap terug zou dan terecht zijn, wat betekent dat precies en wat hebben we daarvoor nodig? En dat beschrijven deze componenten denk ik. Wij waren ons dan bijvoorbeeld bewuster geweest van automatisering en waren wellicht eerder aan de slag gegaan met experimenten.	2, componenten
53	Testen	conflict	Ik denk dat het in het geval van testen voor ons andersom was. Juist omdat we tests hebben geautomatiseerd zijn de handmatige tests afgenomen en daarmee de testinspanning ook. Daarom is het voor ons niet zo'n belangrijke determinant.	2, determinanten
54	Cultuur	steun	Ik denk dat cultuur erg belangrijk is, dat werd bij ons gedreven door het idee dat er echt iets nieuws moest komen en er kwam ook veel nieuw bloed binnen toen.	2, determinanten
55	Cultuur	Steun	De manier van werken had ook impact op de cultuur. Door iedere twee weken	2, determinanten

			iets op te leveren namen de transparantie en het vertrouwen toe.	
56	Nieuw inzicht	Onbewuste keuze	De rol bij de adoptiebeslissing vind ik lastig. Ik denk dat de verandering vooral beargumenteerd is op harde factoren, de voordelen zoals time-to-market. Maar daarachter zat veel meer.	2, determinanten
57	Cultuur	Steun	De verandering kwam juist ook doordat er relatief veel personeelswijzigingen waren in die tijd. Wellicht vond men dat nodig om een verandering in cultuur te bereiken. Ik denk dat het lastig is om oorzaak en gevolg daarbij aan te wijzen, maar het lijkt zeker een relatie te hebben.	2, determinanten
58	Klant	conflict	De klant was bij de beslissing niet zo belangrijk.	2, determinanten
59	Domein	Neutraal	Online front-end is bij ons wel het makkelijkste domein om dit in gang te zetten. Ik denk dat domein meer een hygiënefactor is.	2, determinanten
60	Nieuw inzicht	Onbewuste keuze	Als het er is dan is het oké en als het er niet is heb je een probleem. We waren ons daar niet heel bewust van.	2, determinanten
61	Architectuur	Steun	Voor architectuur zijn wel iets meer bewuste keuzes gemaakt. We hebben daar een aantal non-functional requirements opgesteld die wel van belang zijn. Daarin zie ik wel overlap met CDV. Bij architectuur moet je ook wel, omdat het bepalend is.	2, determinanten
62	Nieuw inzicht	Algemene ontwikkeling	De eisen die de steeds veranderende wereld stelt komen terug in de non-functional requirements en zijn in lijn met CDV. Je wil zaken als loggability and monitorability gewoon hebben, ongeacht CDV. Maar ze helpen ook bij CDV.	2, determinanten
63	Domein	Steun	Het domein kan ook erg beperkend zijn. Een team van ons heeft daar erg mee te maken. We krijgen daar zelfs CI niet aan de praat.	2, determinanten
64	Relatie	determinanten	Dit komt door off-the-shelf software. Deze voldoet dus niet aan de non-functional requirements en er is dus een relatie met architectuur.	2, determinanten

65	Nieuw inzicht	kosten	Geld is wel een factor. Het is niet zozeer een determinant denk, maar eerder een gevolg van deze determinanten. Wij hebben met veel dingen een frisse start gemaakt en daardoor hebben we een vrij moderne architectuur kunnen neerzetten. Maar als ik het bijvoorbeeld heb over het team dat met veel off-the-shelf software werkt, zij kunnen niet zo makkelijk op deze ontwikkeling meeliften. Als ik daar een slag zou willen slaan dan kom ik op kosten uit die ik niet direct kan verantwoorden met de waarde die dat team oplevert. Ik krijg binnen dat domein de business case niet rond. Ik zie dat als een gevolg van de factor domein, maar het speelt een erg belangrijke rol is wat ik voor elkaar kan krijgen.	2, determinanten
66	Nieuw inzicht	organisatoris che factoren	Wat ik nog een beetje mis zit in de organisatorische factoren. Business en IT moeten veel dichter bij elkaar komen. Het team dat software oplevert moet ook mensen uit de business bevatten.	2, determinanten
67	Domein	Steun	Wij hebben ook domeinen met een lagere releasefrequentie, maar hebben het zo ingericht dat ons Java domein onafhankelijk van andere domeinen van releasen. Anders zou je geen CI/CD kunnen toepassen vanwege de afhankelijkheid.	3, status CDV
68	Relatie	determinant en	Anders zou je geen CI/CD kunnen toepassen vanwege de afhankelijkheid. We hebben dat middels loose coupling bijna volledig kunnen elimineren.	3, status CDV
69	Nieuw inzicht	IT push	Dit is vanuit IT bottom-up gepusht. IT vond dat het beter kon. We hadden integratieproblemen en erg veel last van handmatige handelingen.	3, adoptiebeslissing
70	Doelen	Bekend	Door de menselijke handelingen was sprake van een hoge foutgevoeligheid en de business was daar eigenlijk wel klaar met al die fouten. Hierover werd over geklaagd door de business, ook omdat het zoeken en herstellen van fouten lang duurde. Eigenlijk duurde het allemaal te lang en was het onvoorspelbaar.	3, adoptiebeslissing
71	Nieuw inzicht	kosten	Vanuit IT is toen het initiatief gekomen om conform CI/CD te gaan werken, met de overtuiging dat deze investering wel	3, adoptiebeslissing

			terugverdiend zou worden. Er is toen budget beschikbaar gekomen om zelf een volledige CI/CD straat te ontwikkelen.	
72	Testen	Steun	Stap drie was testautomatisering. We hebben daarbij verschillende, ook functionele tests, geautomatiseerd	3, adoptiebeslissing
73	Doelen	Bekend	Nul op de meter, wat betekent geen fouten waar de klant iets van merkt.	3, doelen/voordelen
74	testen	Steun	wij vinden dat je als organisatie geen CDV kan doen zonder continuous testing.	3, doelen/voordelen
75	Doelen	Bekend	Een andere is time-to-market verkorten. Dat is eigenlijk de belangrijkste.	3, doelen/voordelen
76	Doelen	Afwijkend	Een derde is transparantie en traceerbaarheid. Het gaat uiteindelijk om de aantoonbaarheid van onze interne beheersing. Eigenlijk is dat het doel. Transparantie en traceerbaarheid helpen ons bij de aantoonbaarheid van onze interne beheersing.	3, doelen/voordelen
77	Rol voordelen		Eigenlijk hebben al deze voordelen meegespeeld en ik vind dat ze vaak ook erg in elkaars verlengde liggen. Ze hebben alle zeven meegespeeld bij de adoptiebeslissing. We hadden ook hulp van een externe partij. Zij hadden de voordelen vrij scherp aan ons gepresenteerd, dus ik herken eigenlijk volledig wat hier staat.	3, overzicht voordelen
78	Nieuw inzicht	IT push	De betrokken klant heeft eigenlijk geen rol gespeeld bij de keuze voor CI/CD. Dit is later gekomen bij de Agile transitie. Dat was namelijk geen technologische transformatie, maar een bedrijfstransformatie.	3, componenten
79	Rol componenten		De eerste is helder, die hangt samen met korte time-to-market. Automatisering was wel belangrijk. Organisatorische factoren: ….Ik denk dat dit een grote rol speelde	3, componenten
80	Cultuur	Steun	Organisatorische factoren: Wat we in de organisatie zagen is dat de bezetting van mensen hield de manier van werken in stand. Een release iedere drie maanden was prima voor de oude garde. De jongere collega's pikten dit niet en wilden verandering. Daar is de kentering langzaam uit ontstaan. Je zag de buitenwereld veranderen en de	3, componenten

			medewerkers verjongen. Ik denk dat dit een grote rol speelde.	
81	Klant	Neutraal	Ook de kredietcrisis heeft hier aan bijgedragen. Dat heeft zoveel littekens achtergelaten in de financiële sector, je moest wel naar de klant gaan luisteren. Klanten beïnvloeden organisaties namelijk.	3, componenten
82	Nut overzicht		Dit overzicht van componenten had gesprekken destijds wel makkelijker gemaakt. We hebben wel externe hulp gehad, maar een concreet lijstje als deze heb ik nog nooit gezien. Ik had het zeker willen hebben.	3, componenten
83	klant	Neutraal	De rol die de klant speelde is dat we uiteindelijk alles doen voor de klant en bij alles de overtuiging hadden dat het ook echt beter zou worden voor de klant. Dat de klant het niet zou willen is eigenlijk nog nooit bij mij opgekomen. Dit komt natuurlijk ook omdat wij ons bezig houden met webapplicaties. Nieuwe releases uitrollen kan zonder dat de klant daar enige last van ondervindt.	3, determinanten
84	Domein	Steun	Voor domeinen buiten dat van mij zie ik wel dat deze factor bepalend kan zijn. Dus het is echt de vraag of de techniek die je gebruikt, de visie of werkwijze ondersteunt.	3, determinanten
85	Relatie	determinanten	En daarmee kom je eigenlijk uit op architectuur. De link die je legt tussen domein en architectuur is eigenlijk wel terecht.	3, determinanten
86	Architectuur	Steun	Voor ons Java domein geldt dat we middels loose coupling CDV mogelijk konden maken. De eisen die in dit overzicht gesteld worden aan de architectuur hebben we naderhand binnen ons domein kunnen realiseren.	3, determinanten
87	Nieuw inzicht	Onbewuste keuze	Ten tijde van de beslissing waren we ons hier niet zo bewust van. De Java ontwikkelstack hadden we namelijk al ruim voordat we besloten conform CI/CD te gaan werken. In zekere zin is er misschien wel sprake van toeval. Al weet niet wat er was gebeurd als onze ontwikkelstack ongeschikt was gebleken.	3, determinanten
88	Architectuur	Neutraal	Mogelijk is architectuur gewoon een terechte determinant, maar viel dat in	3, determinanten

			ons geval niet zo op, omdat het op orde was.	
89	Cultuur	Steun	Cultuur heeft een heel grote rol gespeeld. Het was een grote investering van de business om zoiets op te gaan zetten. Daarbij is steeds vertrouwen geweest in de professionals en het feit dat het ook iets gaat opleveren. De cultuur van continue verbeteren die is hier gewoon aanwezig.	3, determinanten
90	Cultuur	Steun	Ik denk dat het niet mogelijk is CDV op te zetten zonder dat je bedrijfscultuur hebt van vernieuwing en vertrouwen.	3, determinanten
91	Nieuw inzicht	Onbewuste keuze	Ik denk overigens niet dat iedereen zich daar bewust van was ten tijde van de beslissing. Sommigen wel en anderen zijn hier waarschijnlijk in gevolgd.	3, determinanten
92	Testen	steun	CI/CD kan alleen als geautomatiseerd testen werkt. Het is een randvoorwaarde.	3, determinanten
93	Cultuur	steun	Voor onze organisatie geldt dat onze van product development is een hele zelfstandige en autonome club. En als je dan inzoomt op het echte dev gedeelte, dan is dat daarbinnen ook weer heel autonoom.	4, adoptiebeslissing
94	Nieuw inzicht	IT push	mensen die zelf de strategie bepalen om ook naar de toekomst toe de business te kunnen bedienen. Dus al dit soort keuzes, over hoe willen we werken en waarmee willen we werken, die liggen in het development team.	4, adoptiebeslissing
95	Cultuur	steun	Zij bepalen voor zichzelf wat er nodig is en wanneer. Daar ligt dus gewoon beslissingsbevoegdheid om dit te gaan doen en er komt geen groot besluitvormingsproces aan te pas, behalve dan binnen dat team zelf.	4, adoptiebeslissing
96	Doelen	Bekend	De behoefte tot verbetering kwam voort uit wat fricties. Wij kwamen er gaandeweg achter dat de releases die we deden, inmiddels teruggebracht naar een keer per twee werken, toch nog steeds complex en groot waren.	4, adoptiebeslissing
97	Doelen	Bekend	Daarnaast waren er wat problemen in de OTAP-straten, wanneer verschillende developers daarin werkten.	4, adoptiebeslissing
98	Doelen	Bekend	We deden weleens dinsdag een release en dan waren we de rest van de dinsdagmiddag bugs te fixen. Dat is frustrerend en kan anders	4, adoptiebeslissing

99	Doelen	Bekend	deels vanuit de wens van de business om snel waarde uitgeleverd te krijgen	4, adoptiebeslissing
100	Doelen	Bekend	Met CDV wordt de werkwijze sneller en efficiënter en loop je minder risico.	4, adoptiebeslissing
101	Nieuw inzicht	Algemene ontwikkeling	Wat mogelijk ook een reden was, is dat we met deze werkwijze gewoon met onze tijd en de ontwikkelingen in de wereld meegaan	4, adoptiebeslissing
102	Architect uur	steun	Bij de beslissing hebben we het veel gehad over het geschikt maken van de omgeving voor CDV.	4, adoptiebeslissing
103	Doelen	Bekend	Het gaat erom in staat zijn snel waarde te leveren.	4, doelen/voordelen
104	Doelen	Bekend	Minder risico te lopen en daarmee efficiënter ons product development werk te doen.	4, doelen/voordelen
105	Doelen	Bekend	Je leert ook sneller, zowel op het businessvlak als op het technische vlak.	4, doelen/voordelen
106	Rol voordelen		Ik herken deze natuurlijk wel en denk dat de voordelen die ik genoemd heb hierin wel onder te brengen zijn	4, overzicht voordelen
107	Relatie	Voordelen	Daarnaast zie ik ook wel wat relaties en overlap in deze voordelen. Nummer 2 en 6 hebben best wat met elkaar te maken.	4, overzicht voordelen
108	Nut overzicht		Dit overzicht had onze besluitvorming niet anders kunnen maken.	4, overzicht voordelen
109	Nut overzicht		Bij ons was het dan niet nodig, maar dit overzicht van voordelen kan helpen bij organisaties waar mensen nog overtuigd moeten worden.	4, overzicht voordelen
110	Rol componenten		De eerste herken ik sowieso natuurlijk, dit is waarom we het wilden doen. De tweede speelt natuurlijk een rol. Nummer drie krijgt wel aandacht, we doen veel aan geautomatiseerd testen. Zes is wel een interessante	4, componenten
111	Architect uur	Steun	Dat is eigenlijk een randvoorwaarde om de stap te kunnen maken. Je softwarearchitectuur moet dat wel aankunnen.	4, componenten
112	Nieuw inzicht	Algemene ontwikkeling	Dit komt omdat we over het algemeen wel met onze tijd meegaan en kritisch naar onze architectuur kijken om vast te stellen wat nodig is.	4, componenten
113	klant	neutraal	Zes is wel een interessante. Je moet dit wel uitleggen in het begin. Iedereen was gewend aan tweewekelijkse releases. Zowel de mensen intern als onze gebruikers.	4, componenten

114	Nieuw inzicht	organisatoris che factoren	Organisatorische factoren gaven bij ons geen uitdaging. Wij zijn ook een relatief klein clubje. Ik kan mij voorstellen dat het voor grote organisaties met grote systemen wel een grote uitdaging is, waarbij je veel moet omgooien. Wij werkten al met multidisciplinaire teams die profiteerden van het feit dat we overstapten naar CDV. We zijn er ook langzaam naartoe gegroeid. Als je een transitie van a tot z moet doorlopen dan heb je op organisatorisch vlak wel erg veel te doen.	4, componenten
115	Nut overzicht		Dit lijstje had wel kunnen helpen omdat het een mooi overzicht geeft van wat er nodig is. Bij ons is het niet zo noodzakelijk gebleken, maar anders had ik het graag willen gebruiken hoor.	4, componenten
116	klant	conflict	Ik waag te betwijfelen of een klant inderdaad niet altijd nieuwe software wenst. Ik denk dat de klant daar altijd bij gebaat is. Voor webapplicaties is het wel erg makkelijk en kan een klant zonder last over een nieuwe versie beschikken. Voor andere soorten software kan ik wel voorstellen dat het lastiger is. Maar ook dan moet er waarde zijn voor de klant.	4, determinanten
117	Domein	Steun	Ten aanzien van het domein is het wel zo dat bepaalde onderdelen van ons landschap sterk verouderd zijn. Daarbij is het inderdaad wel lastiger om deze werkwijze van CDV op toe te passen.	4, determinanten
118	Architect uur	Steun	Door technische beperkingen komt het zeker voor dat we voor onderdelen van ons landschap CDV nog niet toepassen.	4, determinanten
119	Cultuur	Steun	Op het gebied van cultuur hebben we wel een transitie achter ons liggen in de afgelopen jaren, wat geleid heeft tot werken met multidisciplinaire teams die het mandaat krijgen dat ze nodig hebben. Dit is wel een proces van jaren. Je moet er met z'n allen in geloven en het management moet het mogelijk maken. Dat is gebeurd. Ik zie het zeker als een hele belangrijke randvoorwaarde.	4, determinanten
120	Testen	Neutraal	Dat testen belangrijk is klopt hoor. Of de testinspanning zeer sterk toeneemt daar twijfel ik aan. Het kan daarvoor ook al erg belangrijk zijn in je werkwijze. Je	4, determinanten

			geautomatiseerde teststraat moet wel op orde zijn.	
121	Nut overzicht		Ik denk dat deze vijf het meeste wel bevatten. Ik kan niets bedenken wat ontbreekt. Dit laatste lijstje, net als de andere, helpen zeker om deze materie in het juiste kader te plaatsen.	4, determinanten
122	Nieuw inzicht	IT push	Toch is het begonnen als een IT feestje. Vanuit IT zijn de eerste stappen gezet.	5, adoptiebeslissing
123	Nieuw inzicht	vooruitlopen of bijblijven	Destijds was het nog een erg nieuwe ontwikkeling en ging het gepaard met de nodige onzekerheid en risico. Toch hadden de CEO en CIO een visie en vonden ze dat we dit moesten doen.	5, adoptiebeslissing
124	Nieuw inzicht	organisatoris che factoren	Enerzijds is het natuurlijk goed voor de organisatie, maar wat ik wel zie is dat het misschien op persoonlijk vlak goed is voor deze mensen. Een interessante ontwikkeling die goed staat op hun cv. Ik denk dat je dat vaker ziet in organisaties	5, adoptiebeslissing
125	Nieuw inzicht	vooruitlopen of bijblijven	Nu zou je het overigens om andere redenen doen. Om bij te blijven bij je concurrenten.	5, adoptiebeslissing
126	Doelen	Bekend	Destijds werd CDV ook als een middel gezien om de marktpositie te vergroten. De overtuiging was dat dat niet ging lukken met de huidige systemen. Daar zaten teveel fouten in en klanten waren niet voldoende tevreden. Althans, de fouten zaten in de softwareopleveringen en dat droeg bij aan klantontevredenheid.	5, adoptiebeslissing
127	Doelen	Bekend	Daardoor ging er veel aandacht uit naar het herstellen en onderhouden van de relatie met de klant. Die aandacht kon niet naar het werven van nieuwe klanten. Daar zit eigenlijk de relatie tussen het marktaandeel vergroten en CDV.	5, adoptiebeslissing
128	Doelen	Bekend	Er was ook wel degelijk een operationeel IT probleem en dat wilde men ook verbeteren. Los van de ambitie om te groeien, was het in stand houden van de bestaande dienstverlening al uitdagend.	5, adoptiebeslissing
129	Doelen	Afwijkend	Daarnaast was er ook een personele overweging. Er waren veel verschillen in de software ten behoeve van de vele verschillende klanten. Kennis hiervoor zat doorgaans in de hoofden van mensen. Als je dat gaat automatiseren, dan haal je die kennis uit de hoofden	5, adoptiebeslissing

			van mensen en komt het bijvoorbeeld in een script terecht.	
130	Rol voordelen		Ik herken deze voordelen allemaal. Ik denk dat we ze goed kunnen ranken en dat ik er net ook wel de meeste heb benoemd.	5, doelen/voordelen
131	Relatie	Voordelen	De eerste en de derde zijn het belangrijkst. De vierde zie ik ook, maar meer als gevolg daarvan	5, doelen/voordelen
132	Relatie	Voordelen	Dus ja, dan kun je ook stellen dat de ontwikkelproductiviteit omhoog gaat en logischerwijs kun je dan ook sneller innoveren.	5, doelen/voordelen
133	Relatie	Voordelen	Voor onze organisatie gold dat al deze voordelen relevant waren, maar ik zie dat vooral terugkomen in de onderlinge relaties.	5, doelen/voordelen
134	Nieuw inzicht	kosten	Bij de adoptiebeslissing is ook wel afgewogen wat de investering kost en wat het zou opleveren. Die oplevering zat in de vraag welke pijnen verminderd zouden worden. En juist daarom vind ik dat verhoogde klanttevredenheid erg zwaar meetelt. We wilden namelijk geen klanten verliezen.	5, doelen/voordelen
135	Nieuw inzicht	kosten	Kostenbesparing op personeel was daar ook een overweging bij, die komt wellicht terug in nummer 5. Bij de onderbouwing van de business case was de personele kostenbesparing wel een belangrijk onderdeel.	5, doelen/voordelen
136	Rol componenten		Snelle en frequente release, zeker. Nummer drie, vier en vijf komen wat mij betreft een beetje bij elkaar. Dit heeft wel gespeeld. De klant speelde wel een belangrijke rol bij de beslissing	5, componenten
137	Nieuw inzicht	Onbewuste keuze	Ik denk dat architectuur niet zo'n belangrijke rol of invloed had in die periode. Dat is nu anders. Het is wel waar wat hier staat, maar ik denk dat dat toen niet zo tussen de oren zat.	5, componenten
138	Architectuur	Steun	Architectuur speelt namelijk wel degelijk een rol en organisaties gaan daar tegenwoordig veel bewuster meer om.	5, componenten
139	Relatie	componenten	Nummer drie, vier en vijf komen wat mij betreft een beetje bij elkaar. Dit heeft wel gespeeld. Ik zie hierin het automatiseren van de delivery pipeline, dus van provisioning tot en met	5, componenten

			deployment. Waarmee geautomatiseerd testen toeziet op kwaliteit.	
140	klant	Steun	De klant speelde wel een belangrijke rol bij de beslissing.	5, componenten
141	Relatie	voordelen	De verhoogde klanttevredenheid moest vooral voortkomen uit de verbeterde kwaliteit en betrouwbaarheid van de release. Niet zozeer door een hogere releasefrequentie.	5, componenten
142	Klant	steun	Sterker nog, een klant moet ook een acceptatieproces in dus het kan ook een nadeel zijn. De software die destijds werd geleverd, was voornamelijk on-premises, dus bij de klant geïnstalleerd.	5, componenten
143	Nut overzicht		Ik denk dat de kennis in dit overzicht wel voor een andere situatie had kunnen zorgen. Destijds was het gedachtegoed nog vrij jong en ik denk dat veel onderdelen nog helemaal niet zo scherp in beeld waren.	5, componenten
144	Nieuw inzicht	Overtuigen	De organisatorisch factoren vallen daarbij het meest op. Het was te lang een IT feestje. Men was vergeten die medewerkers hierin mee te nemen, dus dat moest hersteld worden.	5, componenten
145	klant	steun	Ja herkenbaar, de klant wil niet altijd een hogere releasefrequentie	5, determinanten
146	Nieuw inzicht	Onbewuste keuze	We zijn begonnen in de domeinen midoffice en backoffice. Hier was men destijds wel bewust van, maar wist niet voldoende welke impact daaraan verbonden was.	5, determinanten
147	Domein	steun	Toen we bij het backoffice domein uitkwamen, met veel meer batchverwerkingsprocessen bleek het wel ingewikkeld en ontstond de vraag hoe dit te doen	5, determinanten
148	Nieuw inzicht	Onbewuste keuze	En ook al weet je dat je dergelijke problematiek gaat raken, denk ik dat er weinig bewustzijn op is geweest.	5, determinanten
149	Relatie	determinanten	Want het domein heeft, samenhangend met de architectuur daar heel veel invloed op.	5, determinanten
150	Architectuur	steun	De architectuur heeft erg vaak in de weg gezeten bij het implementeren van CDV. Vooral in het backoffice domein, door de batchverwerkingsprocessen.	5, determinanten
151	Architectuur	steun	Het is wel voorgekomen dat we destijds voor bepaalde applicaties CDV nog maar even niet hebben toegepast.	5, determinanten

152	Cultuur	steun	Ik denk dat cultuur een van de belangrijkste factoren is om rekening mee te houden.	5, determinanten
153	Cultuur	Steun	Verschillen binnen een organisatie zijn hierbij wel waarneembaar. In het ene team of subteam kan een andere cultuur bestaan dan in het andere. Dat gaat ook organisch, dezelfde soorten mensen zoeken elkaar toch een beetje op. Een groep mensen kan erg openstaan voor verandering en innovatie, terwijl een andere groep erg behoudend is	5, determinanten
154	Nieuw inzicht	IT push	Een ontwikkelaar zal eerder willen innoveren dan een beheerder.	5, determinanten
155	Cultuur	Steun	Er is ook wel rekening gehouden met de factor cultuur bij het nemen van de adoptiebeslissing. Daarbij is heel bewust gekozen om het tegelijk te introduceren met Agile werken.	5, determinanten
156	Testen	Steun	Testen heeft absoluut invloed gehad op de adoptiebeslissing. Waar het opleveren van een server eerst bijvoorbeeld een week duurde, met fouten. Was het na het traject met een druk op de knop na vijftien minuten voor elkaar, foutloos.	5, determinanten
157	Testen	Steun	Het is vooral een factor omdat je niet moet onderschatten hoeveel werk testautomatisering is.	5, determinanten
158	Nieuw inzicht	Greenfield vs. Bestaand	Bij een nieuw systeem is dat niet zo'n probleem want dan groeit het mee. Voor een bestaand systeem moet je eerst een hele berg verzetten, je hebt echt een enorme achterstand.	5, determinanten
159	Relatie	determinanten	Vanwege de factor testen hebben we wel besloten om voor applicaties nog even niet de CDV werkwijze te hanteren. Dit is eigenlijk hetzelfde als ik hiervoor gezegd hebt over de backoffice applicaties. Vanwege de architectuur dus.	5, determinanten
160	Nut overzicht		Ik denk dat deze determinanten een rol hebben gespeeld bij de beslissing. Wat meer bewustzijn had zeker kunnen helpen. Het is een fijn lijstje dat je kunt hanteren	5, determinanten
161	Nut overzicht		Ik zie niet een determinant die mist in dit overzicht.	5, determinanten
162	Nieuw inzicht	Agile enabler	Daarmee creëerden we een snelheid waarvoor een verregaande automatisering nodig is. Je loopt	6, status CDV

			bijvoorbeeld aan tegen de grenzen van je testcapaciteit.	
163	Klant	neutraal	Dus toen zijn we met CI gestart als facilitator van Agile werken en om tegemoet te komen aan de flexibiliteit die klanten van ons vroegen.	6, status CDV
164	Nieuw inzicht	Agile enabler	Om van Agile werken een succes te maken moesten we CI gaan doen.	6, status CDV
165	Nieuw inzicht	Algemene ontwikkeling	Het is niet zo dat we nog niets deden, we hadden delen al wel geautomatiseerd. Later hebben we in bredere context gekeken naar CI en hebben we het fenomeen volledig toegepast,	6, status CDV
166	Nieuw inzicht	Onbewuste keuze	De stap naar CI kwam aanvankelijk misschien enigszins onbewust.	6, status CDV
167	Nieuw inzicht	Algemene ontwikkeling	Het zoveel mogelijk genereren van software, zoveel mogelijk controleren van standaarden en richtlijnen, geautomatiseerde tests, het geautomatiseerde proces om omgevingen op te tuigen, dat zijn dingen die je in een waterval ook prima kunt gebruiken. Elke professionele organisatie is met dat soort dingen bezig.	6, status CDV
168	Nieuw inzicht	Onbewuste keuze	Toen wij naar een ritme van twee weken wilden, zijn we ons gaan verdiepen in de principes die daarover worden afgesproken in de wereld om ons heen. Dat blijkt dan CI te heten.	6, status CDV
169	klant	neutraal	Daarbij wilden we meer met klanten samenwerken, we wilden duidelijker prioriteren en ook samen met klanten prioriteren	6, status CDV
170	Doelen	Bekend	We wilden ook de kwaliteit verhogen	6, status CDV
171	Doelen	Bekend	flexibeler zijn, wat betekende tussentijds prioriteiten wijzigen	6, status CDV
172	Nieuw inzicht	Overtuigen	Die beslissing hebben we uiteindelijk samen met de medewerkers gemaakt. Je moet daarover wel in gesprek gaan. Je kunt dat niet zomaar afdwingen als management.	6, status CDV
173	klant	Steun	We zijn ook wel in staat om software iedere twee weken aan de klant te releasen, maar klanten willen dat over het algemeen niet.	6, status CDV
174	doelen	Bekend	In het midden van het proces echter, bleek toch ook wel dat wij niet opleverden wat gevraagd werd en is het aan ons. We zijn toen gaan kijken hoe	6, status CDV

			we het development proces konden versnellen.	
175	Klant	neutraal	We wilden anders met klanten omgaan	6, status CDV
176	doelen	Bekend	intern sneller werken	6, status CDV
177	Nieuw inzicht	Agile enabler	Daarnaast zagen we de wereld om ons heen steeds sneller veranderen en wilden we flexibeler zijn om daar tegen bestand te zijn.	6, status CDV
178	Nieuw inzicht	Agile enabler	Daarom is besloten om over te gaan tot Agile werken. CI is later het middel gebleken om dat voor elkaar te krijgen en die snelheid te bereiken.	6, status CDV
179	Nieuw inzicht	verschil klant/leverancier	Vervolgens hebben we besloten om volledig tot CI over te gaan. Als je CDV ziet als het grotendeels geautomatiseerd samenstellen op opleveren van een release, dan denk ik dat we dat voor 95% hebben bereikt.	6, status CDV
180	klant	Steun	Als je daarin meeneemt dat het ook daadwerkelijk op een productieomgeving terecht moet komen, dan zijn we daarin afhankelijk van onze klanten.	6, status CDV
181	klant	steun	De vraag is of je dit voor dergelijk mission critical systeem moet willen. En dat is een overweging die vooral aan de klant is.	6, status CDV
182	Nieuw inzicht	verschil klant/leverancier	Wat lastig is, is dat klanten niet willen dat wij toegang hebben tot hun data. Klanten zouden wel kunnen leren van onze ervaring met betrekking tot automatisch testen en deployen	6, status CDV
183	Nieuw inzicht	verschil klant/leverancier	Als ik droom over de toekomst zie ik wel voor me dat klanten nieuwe versies naar hun acceptatieomgeving gepusht krijgen en dat wij een sein terugkrijgen als de acceptatietest succesvol is verlopen. Vervolgens zouden we dan automatisch kunnen releasen naar productie.	6, status CDV
184	doelen	Bekend	Vroegtijdige feedback van klanten, waardoor we de juist dingen bouwen.	6, doelen/voordelen
185	doelen	Bekend	De mogelijkheid om flexibel te kunnen ingrijpen bij veranderende prioriteiten.	6, doelen/voordelen
186	doelen	Bekend	Snelheid. Ik denk dat dat de belangrijkste zijn.	6, doelen/voordelen
187	Rol voordelen		Ja, ik herken ze allemaal wel. Deze argumenten hebben we ook wel gebruikt om dit aan de man te krijgen.	6, doelen/voordelen
188	Architectuur	conflict	Voor wat betreft de tweede, ons softwareproduct bestaat al een tijdje,	6, componenten

			maar het is niet zo dat we onze complete softwarearchitectuur overhoop hebben moeten halen om dit te realiseren.	
189	Architectuur	conflict	Onze keuzevrijheid hierin is beperkt, omdat dingen met elkaar kunnen samenhangen. Ik weet niet of dit komt door verouderde architectuur, het heeft ook met de complexiteit van het systeem te maken.	6, componenten
190	Architectuur	conflict	Met een systeem dat zo complex is als het onze, los van de architectuur is flexibiliteit moeilijk bereikbaar. Als je iets bouwt dat volledig op zichzelf staat is er niet zoveel aan de hand, maar wanneer je te maken hebt met kritieke verwerkingen dan moet je steeds controleren of alles nog goed werkt.	6, componenten
191	cultuur	Steun	Nummer tien, deze transitie heeft wel gezorgd voor commitment, eigenaarschap en enthousiasme bij medewerkers. We hadden daar ook erg op gehoopt. We hebben deze verandering zorgvuldig moeten aanpakken. We zijn namelijk een development club. We kunnen het ons niet permitteren om bij dergelijke verandering bijvoorbeeld de helft van onze mensen te verliezen.	6, componenten
192	Rol componenten		Samenvattend: 1, 6, 9 en 10 zijn de belangrijkste geweest.	6, componenten
193	klant	Steun	De klant is inderdaad een erg belangrijke determinant, vooral in positieve zin. Ze waarderen de versnellingen die we hebben gemaakt,	6, determinanten
194	klant	Steun	maar de laatste stap is niet altijd gewenst. Klanten willen toch nog zelf testen en ook vooral hun eigen inrichting testen. Voor de laatste stap naar CDV waarbij je ook automatisch kunt deployen naar de productieomgeving is de klant op dit moment de beperkende factor.	6, determinanten
195	domein	Steun	Dus als je dat allemaal tot het domein rekent, dan is dat zeker medebepalend voor de drempel die is er is om echt aan CDV te doen.	6, determinanten
196	Architectuur	neutraal	Ondanks deze eigenschappen waren we uitstekend in staat te komen tot waar	6, determinanten

			we nu zijn en zit de architectuur niet in de weg bij een CDV implementatie.	
197	Architectuur	Steun	In het algemeen denk ik het wel dat architectuur een determinant is.	6, determinanten
198	Cultuur	Steun	Ja, ik denk wel dat cultuur een determinant is. Vanuit mijn rol als manager ben ik erg doordrongen van wat hier staat.	6, determinanten
199	Cultuur	Steun	Ik probeer invulling te geven aan dienend leiderschap en vooral faciliterend te zijn	6, determinanten
200	Cultuur	Steun	Als management moet je dit wel aandurven. Als het hoger management niet accepteert dat je die teams het vertrouwen en de zelfstandigheid geeft, dan is het eigenlijk kansloos.	6, determinanten
201	Nieuw inzicht	organisatorische factoren	Nu is iedereen bij ons developer en moet in ieder geval ook kunnen programmeren. Niet iedereen hoeft de grootste development expert te zijn, maar voor iemand die alleen maar kan analyseren of testen is geen plek meer.	6, determinanten
202	Testen	Steun	Verder ben ik het eens met wat hier staat. Je moet vooraf zeker bewust zijn van het feit dat je geautomatiseerd testen nodig hebt, anders ga je dit niet realiseren.	6, determinanten
203	Nut overzicht		Ik herken alle vijf deze determinanten. Ik denk dat je over alle vijf heel goed moet nadenken of je deze stap kunt zetten.	6, determinanten
204	Nut overzicht		We hebben deze determinanten niet zo expliciet onderkend en stuk voor stuk beoordeeld, maar nu ik ze zo zie staan denk ik wel dat het goed zou zijn geweest om er op die manier naar te kijken.	6, determinanten
205	Nieuw inzicht	organisatorische factoren	De reguliere managementrollen verdwijnen. Dat is een belangrijke organisatorische factor. Je zou dat kunnen zien als onderdeel van cultuur, maar misschien staat het wel op zichzelf. Het organogram wijzigt gewoon en daar moet je wel toe bereid zijn.	6, determinanten